(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 May 2001 (03.05.2001)

PCT

(10) International Publication Number
WO 01/31882 A1

(51) International Patent Classification[7]: H04L 29/06, 12/56

(21) International Application Number: PCT/US00/28370

(22) International Filing Date: 12 October 2000 (12.10.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/425,744 22 October 1999 (22.10.1999) US

(71) Applicant: VITESSE SEMICONDUCTOR CORPORATION [US/US]; 741 Calle Plano, Camarillo, CA 93012 (US).

(72) Inventors: VARMA, Anujan; 519 Hagar Court, Santa Cruz, CA 95064 (US). KUMAR, Praveen, D.; 5008 Amberwood Drive, Fremont, CA 94555 (US).

(74) Agent: OLYNICK, Mary, R.; Beyer Weaver & Thomas, LLP, P.O. Box 778, Berkeley, CA 94704-0778 (US).

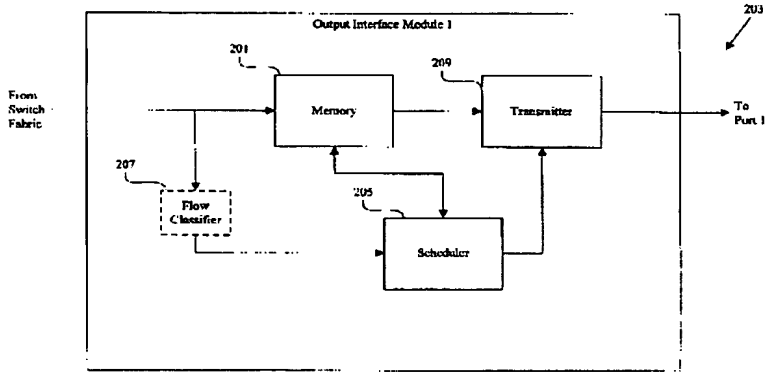
(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
With international search report.

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHODS AND APPARATUS FOR SCHEDULING PACKET TRANSMISSION AT A NETWORK PORT



(57) Abstract: Disclosed are apparatus and methods for scheduling packets belonging to a plurality of flows on a transmission link so as to provide bandwidth guarantees to each flow. The flows are serviced in a round-robin manner, where each round consists of the transmission of a sequence of packets from flows that are eligible to transmit in the corresponding round. In general terms, data structures are created and provided, and the data structures associate eligible flows with each round. When a particular flow has exceeded its bandwidth allocation temporarily, the round in which the flow becomes eligible to transmit its next packet is calculated and the flow is inserted within a service queue associated with that round. The scheduling methods utilize these data structures to then transmit eligible flows during each round without calculating whether each flow is eligible in each round. Likewise, when a packet arrives from a flow that has been idle, the round in which the arrived packet is eligible for transmission is calculated immediately and, consequently, its eligibility is then not checked during each round.

WO 01/31882 A1

METHODS AND APPARATUS FOR SCHEDULING PACKET TRANSMISSION AT A NETWORK PORT

BACKGROUND OF THE INVENTION

The present invention relates generally to switching packets through output ports in a packet switch or other network device. Specifically, it relates to devices for scheduling and determining the order of the packets transmitted through an output port.

Figure 1 is a diagrammatic representation of a switching device 100. The switch device 100 has a plurality of input ports 105 (e.g. input ports 1 through N) and a plurality of output ports 107 (e.g. output ports 1 through N). The switch device 100 generally receives packets on a particular input port and routes these received packets to one of the output ports 107.

Each input port is coupled with an input interface 101 that transmits the received packet to a switch fabric 102. The switch fabric 102 works in conjunction with the input interfaces 101 to transmit the received packet to a selected output interface 103. For example, a packet is received on input port 1 (105a) and passed through input interface 1 (101a) and switch fabric 102 to output interface 1 (103a). Each output interface 103 then determines when to transmit the received packet onto the associated output port 107. For example, output interface 1 (103a) schedules and transmits data to output port 107a. In sum, each output interface 103 schedules received packet for transmission on the associated output port.

Each packet is associated with a flow, session, or conversation. Typically, there are several packets associated with each flow. Thus, the packets are grouped into flows or flow queues. In other words, each flow will be associated with a queue of packets at the output interface 103. It is the job of the output interface 103 to schedule data from the plurality of flow queues on the associated output port 107.

Several techniques are currently available for scheduling packet transmissions from various flow queues. One technique is to referred to as "round-robin" scheduling. The round-robin scheduler goes through the flows one by one and transmits a packet from each flow. For example, a packet is transmitted from flow 1; a packet is transmitted from flow 2; etc. If a flow's queue is empty, that flow is skipped, and a packet is transmitted from the next non-empty flow.

Although the round-robin scheduler is adequate for some applications, sometimes variable bandwidths are desired for different flows. For example, a particular flow may require a relatively high throughput compared to other flows. In this case, the scheduler must allocate more bandwidth to the high-bandwidth flow relative to the other flows. The simple round-robin scheduler can not allocate such unequal bandwidths to the different flows. Additionally, when the packet size is not fixed, the simple round-robin technique may result in particular flow taking too much bandwidth because its packets have a relatively large size.

Another scheduling technique that partially solves the above-described problem is the weighted round-robin scheduler. In this approach, each flow queue is assigned a weight value. During each round, each flow queue may transmit a designated number of packets based on the assigned weight of that flow. For

example, a flow that has an associated weight of 5 packets may transmit 5 packets of data per round, while a flow that has a weight of 2 packets may transmit 2 packets of data per round. Although the weighted round-robin scheduler provides more flexibility than the simple round-robin scheduler, the weighted round-robin approach still does not allow precise control of the bandwidth allocation when the packets are not of the same size.

A scheduling algorithm that extends the round-robin approach to variable-size packets is the deficit round-robin technique. This approach is further described in "Efficient Fair Queuing Using Deficit Round-Robin" by M. Sreedhar and George Varghese, IEEE/ACM Transactions on Networking, Volume 4, Number 3, June 1996, pages 375-385, which is incorporated herein by reference in its entirety. This technique works as follows: Each flow is allocated a certain weight in terms of the number of bytes the flow may transmit in each round. However, since the packet size is variable, this weight may not correspond to a whole number of packets available for transmission in the flow's queue during a given round. For example, a particular flow might have an assigned weight of 500 bytes per round. If the first two packets in the flow's queue have sizes 300 and 250, respectively, transmitting both packets in the current round will result in the flow's bandwidth usage exceeding its allocation by 50 bytes in the current round. The deficit round-robin technique, in this example, allows only the first packet of the flow to be transmitted in the current round, with the remaining allocation of (500-300) - 200 bytes to be carried over to the next round in the form of a deficit.

The deficit round-robin technique maintains a deficit value for each flow

queue. The deficit value indicates the amount of unused bandwidth of the flow from the previous round, which can be used by the flow in the current round. During a given round, the maximum number of bytes that a flow is allowed to transmit is determined by adding its deficit value to the weight of the flow. This sum is stored in a counter. As packets are transmitted from the flow's queue, this counter is decreased by the size of each transmitted packet. The flow is allowed to transmit packets as long as the counter value is greater than or equal to the size of the next packet in the flow's queue waiting for transmission. If the former becomes less than the latter, the flow can no longer transmit packets in the current round. The leftover value of the counter is carried over to the next round as the deficit value of the flow. This ensures that, over a time interval spanning several rounds, each flow receives, on the average, a bandwidth allocation approximately equal to its assigned weight, although it might receive more or less than its weight in a given round.

Although the deficit round-robin approach allocates transmission bandwidth accurately even when the packets are of variable size, it has its own associated disadvantages: First, in order for the scheduler to determine if a packet at the head of a flow's queue is eligible for transmission in the current round, the size of the packet must be known. This size information may not be available to the scheduler. For example, the entire packet may not have been received and stored in memory by the time the scheduling decision needs to be made. In that case the packet will have to wait until the next round, incurring additional delays.

A second drawback of the above deficit round-robin scheduling technique is that the computations involved in checking the eligibility of packets during the

scheduling process can cause some of the bandwidth to be wasted when the number of flows being scheduled is large. During each round, the scheduler determines the eligibility of a packet at the head of a flow's queue for transmission by adding the current deficit of the flow to its weight and comparing the resulting sum to the size of the packet. If the sum is less than the size of he packet, the scheduler must skip the corresponding flow and proceed to the next flow, performing the same computations on it. During a given round, the scheduler may have to skip several flows before finding an eligible packet to schedule. This may cause the transmission link to remain idle periodically during the round while the scheduler is performing its computations, thus wasting a part of the bandwidth capacity of the link. Thus, the deficit round-robin approach does not scale well to a large number of flow queues.

In view of the foregoing, there is a need for inproved methods and appartuses for scheduling data output.

SUMMARY OF THE INVENTION

Broadly speaking, the present invention fills these needs by providing apparatus and methods for scheduling flows for transmission during a selected round prior to transmission of such flows of such round. In general terms, data structures are created or provided, and the data structures associate eligible flows with each round. For example, when a particular flow has a deficit and is no longer allowed to transmit data, a reference to that particular flow is inserted within a data structure that is associated with the round that may next transmit the particular flow. That is, the reference to the particular flow is associated with the future round in which the particular flow can be transmitted because it's deficit is predicted to fall to zero or below at that future round.

In one embodiment, the invention pertains to a method for scheduling a data portion for transmission through an output port. The data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, and the data portion is associated with a flow. A round number is selected from the plurality of round numbers for transmitting the data portion through the output port, and the flow associated with the first data portion is associated with the selected round number such that the first data portion is automatically transmitted during the selected round number without first calculating a deficit or surplus for the flow associated with the first data portion.

In an alternative embodiment, when the flow associated with the first data portion contains other data portions or the flow does not have a deficit, the round number is selected as a round number that is currently transmitting data. In yet

another embodiment, when the flow associated with the first data portion is empty and has a deficit, the round number is selected as a later round number in which the first data portion becomes eligible for transmission.

In another method embodiment, a method for maintaining a data structure that is used to schedule transmission of a plurality of data portions during a plurality of sequentially numbered rounds is disclosed. The data structure includes a plurality of service queues that are each associated with a one of the round numbers, and each of the service queues also are associated with one or more flows. Each associated flow is eligible for transmission when its associated round number is reached.

A current data portion associated with a current flow is received. The current flow is also associated with a current deficit counter and a current credit counter. The current credit counter is maintained to indicate whether the current flow has a deficit and is ineligible to transmit. The current deficit counter is maintained to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission. When the current data portion's associated flow is not associated with one of the service queues, a round number in which the current flow is eligible for transmission is determined based on the current deficit counter and associating the current flow with the service queue of the determined round number.

In another embodiment, the invention pertains to an apparatus for scheduling a data portion for transmission through an output port. The data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, wherein the data portion is associated with a flow. The apparatus includes a storage

device arranged to receive the data portion and a database arranged to associate a round number with a flow in which data portions from the flow may be transmitted. The apparatus also includes a scheduler arranged to select a round number from the plurality of round numbers for transmitting the data portion through the output port and to assign the flow associated with the first data portion to the selected round number via the database. The apparatus further includes a transmitter arranged to automatically transmit the data portion during the selected round number without first calculating a deficit or surplus for the flow associated with the first data portion.

In another apparatus embodiment, an apparatus for maintaining a data structure that is used to schedule transmission of a plurality of data portions during a plurality of sequentially numbered rounds is disclosed. The data structure includes a plurality of service queues that are each associated with a one of the round numbers, and each of the service queues also is associated with one or more flows. Each associated flow is eligible for transmission when its associated round number is reached. The apparatus includes a storage device arranged to receive data portions and the structure, and a scheduler arranged to receive a current data portion associated with a current flow associated with a current deficit counter and a current credit counter. The scheduler is further arranged to maintain the current credit counter to indicate whether the current flow has a deficit and is ineligible to transmit, maintain the current deficit counter to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission, and if the current data portion's associated flow is not associated with one of the service queues, determining a round number in which the current flow is eligible for transmission based on the current deficit counter and associating the

current flow with the service queue of the determined round number.

In an alternative embodiment, the invention pertains to a computer program product arranged to cause a computer to schedule a data portion for transmission through an output port. The data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, and the data portion is associated with a flow. The computer program product includes computer code that selects a round number from the plurality of round numbers for transmitting the data portion through the output port and computer code that associates the flow associated with the first data portion with the selected round number such that the first data portion is automatically transmitted during the selected round number without first calculating a deficit or surplus for the flow associated with the first data portion. The computer program product further includes a computer readable medium that stores the computer codes.

In another embodiment, a computer program product arranged to cause a computer to maintain a data structure that may be used to schedule transmission of a plurality of data portions during a plurality of sequentially numbered rounds is disclosed. The data structure includes a plurality of service queues that are each associated with a one of the round numbers. Each of the service queues also is associated with one or more flows, and each associated flow is eligible for transmission when its associated round number is reached. The computer program product includes computer code that receives a current data portion associated with a current flow associated with a current deficit counter and a current credit counter and computer code that maintains the current credit counter to indicate whether the current

flow has a deficit and is ineligible to transmit. The computer program product also includes computer code that maintains the current deficit counter to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission, computer code that determines a round number in which the current flow is eligible for transmission based on the current deficit counter and associating the current flow with the service queue of the determined round number when the current data portion's associated flow is not associated with one of the service queues, and a computer readable medium that stores the computer codes.

In yet another embodiment, a data structure for scheduling transmission of a plurality of data portions during a plurality of sequentially numbered rounds is disclosed. The data structure includes a plurality of service queues that are associated with one or more flows and a plurality of service queue references that are each associated with one of the round numbers and a one of the service queues. The associated flows of each service queue are eligible for transmission when its associated round number is reached.

These and other features and advantages of the present invention will be presented in more detail in the following specification of the invention and the accompanying figures which illustrate by way of example the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be readily understood by the following detailed description in conjunction with the accompanying drawings, wherein like reference numerals designate like structural elements, and in which:

Figure 1 is a diagrammatic representation of a switching device.

Figure 2 is a diagrammatic representation of an output interface module in accordance with one embodiment of the present invention.

Figures 3A and 3B are a diagrammatic representation of the memory of Figure 2 in accordance with one implementation of the present invention.

Figure 4 is a flowchart illustrating a process that is executed upon packet arrival in accordance with one embodiment for the present invention.

Figure 5 illustrates a flowchart of a process of performing a deficit counter scan to update the deficit counters of idle flows in accordance with one embodiment of the present invention.

Figure 6 is a flowchart illustrating the operation of Figure 5 for updating the deficit counter of the current flow during the scan cycle in accordance with one embodiment of the present invention.

Figure 7 illustrates the flowchart of the operation of Figure 4 for selecting the round number for insertion of the flow ID of the arrived packet in accordance with one embodiment of the present invention.

Figure 8 shows a timeline of rounds that were in progress, are in progress, and are scheduled for process in accordance with one embodiment of the present invention.

Figure 9 is a flowchart illustrating the operation of Figure 4 of inserting the flow ID of the arrived packet in accordance with one embodiment of the present invention.

Figures 10A through 10C illustrate an example of implementing the process of Figure 9 to insert a flow 5 within the service queue for round 3 in accordance with one embodiment of the present invention.

Figures 11A and 11B are flowcharts illustrating the process that occurs after a current packet (*e.g.*, an arrived packet of Figure 4) completes transmission in accordance with one embodiment of the present invention.

Figure 12 is a flowchart illustrating the operation of Figure 11 for setting the deficit counter of a flow when the flow goes idle in accordance with one embodiment of the present invention.

Figure 13 is a flowchart illustrating the operation of Figure 11B of calculating the round number in accordance of one embodiment of the present invention.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

Reference will now be made in detail to specific embodiments of the invention. While the invention will be described in conjunction with specific embodiments, it will be understood that it is not intended to limit the invention to the described embodiments. On the contrary, it is intended to cover alternatives, modifications, and equivalents as may be included within the spirit and scope of the invention as defined by the appended claims. In the following description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. The present invention may be practiced without some or all of these specific details. In other instances, well known process operations have not been described in detail in order not to unnecessarily obscure the present invention.

Figure 2 is a diagrammatic representation of an output interface module 203 in accordance with one embodiment of the present invention. The output interface module may be arranged with other components to form a switch device (as described in Figure 1). As shown, the output interface module 203 includes memory 201, transmitter block 209, scheduler 205, and flow classifier 207. Packets are received from the switch fabric into memory 201 and flow classifier 207.

The flow classifier 207 is optional and determines the flow ID of the received packet. Alternatively, this identification function is performed by the scheduler 205. The scheduler 205 determines which packet to transmit next after the transmitter 209 transmits the currently scheduled packet. Generally, the scheduler 205 guarantees quality of service by determining when to transmit a flow and how many packets from that flow to transmit.

The memory 201 may be configured in any suitable manner for storing the arriving packets from multiple flows and storing data structures used by the scheduler to track and schedule flows. For example, whole packets may be stored contiguously, and each packet may have an associated link to the next packet. Unfortunately, a contiguous configuration may lead to fragmentation problems since each memory address may be filled with packets of varying sizes. Alternatively, the packets may be divided into equally sized units (e.g. 64 bytes in length).

Figures 3A and 3B are a diagrammatic representation of the memory 201 of Figure 2 in accordance with one implementation of the present invention. The memory 201 includes a packet memory 301, a plurality of per flow tail pointers 303, a plurality of per flow head pointers 309, a plurality of service queue head pointers 305, and a plurality of service queues 307. Each of the per flow tail pointers references an end packet of a particular flow. As shown in Figure 3A, the per flow tail pointer 303a references the end packet of flow 0 (*i.e.*, packet 1) within the packet memory 301. The per flow tail pointer 303b references the end packet of flow 1 (*i.e.*, packet 60) within packet memory 301. The per flow tail pointers provide a convenient mechanism for adding a newly arrived packet to end of a flow. That is, the appropriate flow's per flow tail pointer is obtained, and the obtained per flow tail pointer is altered to reference the newly arrived packet.

Each of the per flow head pointers reference a starting packet of a particular flow. As shown in Figure 3A, the per flow head pointer 309a references packet 0 of the flow that includes packets 0 and packet 1, and the per flow head pointer 309b references packet 2 of the flow includes packets 2 through packet 60. The per flow

head pointers provide mechanisms for transmitting packets of a particular flow starting at the first packet. The head pointer of a particular flow is retrieved and its associated packet is transmitted. If the remaining packets of the flow are linked together or consecutive, the rests of the packets for the particular flow may then be sequentially obtained and transmitted. Of course, other arrangements may be implemented for retrieving and/or adding packets from and to the packet memory.

The scheduler selects packets for transmission from the flow queues by examining the flow queues that are eligible to transmit packets in a round-robin manner. The sequence of flows that are eligible to transmit packets during a given round are maintained in a service queue associated with the corresponding round. As shown in Figure 3B, an array of service queue head pointers 305 is used to reference the first flow within a service queue associated with each round. The flows within the referenced service queue are eligible for transmission during the round that is associated with the head pointer 305. For example, service queue head pointer 305a for round 3 references a service queue that begins with flow 2 and ends with flow 21.

The service queue head pointer array 305 contains a total of k pointers, pointing to the service queues of k consecutive rounds, where k is a suitable parameter chosen in the implementation. The pointers are re-used cyclically. That is, on completion of the service queue referenced by the last (k-1)th pointer, the next round begins at the service queue referenced by head pointer 0 again.

The variables associated with the flows are maintained in an array of flow records 307. Each flow record stores the flow ID of the associated flow and the variables associated with the flow. These variables include: (i) a flow weight that

indicates the relative bandwidth share of the flow, represented as the number of bytes the flow is allowed to transmit per round; (ii) a credit counter that maintains the unused bandwidth the flow has accumulated up to the current round; (iii) a deficit counter that keeps track of the number of rounds the flow must wait before it becomes eligible for service again; and (iv) an idle bit to indicate if the associated flow has one or more packets waiting in its queue. The idle bit is set to 1 when the associated flow has no packets in its queue. Each flow record also contains a next flow pointer to point to the next flow in the same service queue when the flow is part of a service queue. Each service queue head pointer 305 references the flow record of the first flow in the corresponding service queue; the subsequent flow records of flows belonging to the same service queue can be accessed by following the next flow pointers in the flow records.

Each service queue includes one or more flows whose packets may be transmitted, in turn, during a particular round. The flows within a service queue are linked via next flow pointers. For example, the service queue head pointer 305a for round 3 references flow 2, which references flow 3; which references flow 4; which references flow 21. Thus, packets from flows 2, 3, 4, and flow 21 may be transmitted during round 3.

During a round, the associated service queue head pointer 305 is first obtained. The obtained head pointer 305 is associated with a flow ID of the first flow that is eligible for transmission within the associated service queue 307. Thus, the first flow's flow ID is then obtained. The first packet associated with the flow ID may then be obtained from the per flow head pointers 309. A next packet associated with

the same flow ID may be obtained via the first packet's next packet pointer, or the next packet may simply be positioned within the next memory location after the first packet. Alternatively, the next flow within the current round's service queue 307 may be obtained via the next flow pointer of the first obtained flow ID within service queues 307.

The service queues 307 and service queue head pointers 305 provide a convenient mechanism for indicating prior to transmission which flows are to be transmitted for a particular round. In other words, the present invention provides look-ahead mechanisms that can calculate the next round in which each flow may transmit data. Specialized data structures (*e.g.*, service queues 307 and service queue head pointers 305) allow rapid transmission of a particular set of flows during a particular round without checking each flow during the round in progress to determine whether it can send data during such round in progress.

These data structures may be constructed by any suitable processes. For example, when a flow becomes idle after it has a deficit, a next round in which the flow may transmit is then calculated based on the deficit and the amount of data the flow is allowed to transmit during a single round. For example, if a flow obtains a deficit of 500 bytes during round number 1 and is allowed to transmit 400 bytes of data per round, the flow can again transmit during round number 3, but not during round number 2. Several embodiments for constructing the services queues 307 and service queue head pointers 305 are described below in reference to Figure 4 though 13. Additionally, these data structures may be updated at any suitable frequency. For example, the updating may be event driven, wherein an update occurs after a packet

arrives, a packet is transmitted, and/or a flow becomes idle. By way of another example, the updating process may occur at regular time intervals (*e.g.*, while a flow is idle).

The initial values of the variables in each flow record 307 may be set utilizing any suitable mechanism. In the illustrated embodiment, the flow weight of a flow is a constant and is set based on the relative bandwidth share of the corresponding flow. The flow weight is specified in terms of the number of bytes the corresponding flow is allowed to transmit per round. For example, if flow 1 has a weight of 500 bytes per round and flow 2 has 1000 bytes per round, then flow 2 is assigned twice the bandwidth share compared to flow 1. The credit counter is initially set equal to the corresponding flow weight and the deficit counter is set to zero. All the idle flags are set to 1 to indicate that no packets are queued in the system. Finally, all service queue head pointers are set to null values to indicate that there are no packets eligible for scheduling.

Figure 4 is a flowchart illustrating a process 400 that is executed upon packet arrival in accordance with one embodiment for the present invention. When a packet first arrives (*e.g.*, within the output interface 203 of Figure 2), it is added to the packet memory (*e.g.*, packet memory 301 of Figure 3). If the packet's flow is not referenced within a round number's service queue (*e.g.*, 307 of Figure 3), a round number is then determined for inserting the flow ID of the arrived packet, and the packet is inserted within the service queue of the selected round.

As shown, the flow ID is first determined for the current packet in operation 401. Next, the packet is added to the packet memory at the end of the flow in

operation 403. Of course, the packet may be inserted within any empty location within the packet memory so long as the packet is associated with its corresponding flow. Generally, if the packet is the first packet within a particular flow, the packet is simply inserted within the packet memory at the end of the previous flow.

After the packet is added to the packet memory (e.g. 307 of Figure 3), the tail pointer associated with the flow queue of the inserted packet is updated in operation 405. In other words, the tail pointer references the new end of the flow (i.e., the newly added packet). In operation 407, it is determined whether the flow queue is empty upon the packet's arrival. In other words, it is determined whether the current packet represents the start of the flow queue. If the flow queue is not empty, the packet arrival process 400 ends since the packet's flow is already referenced within an associated service queue.

If the flow queue was previously empty, a round number for insertion of the flow ID within the corresponding service queue is selected in operation 408. In other words, the flow is scheduled for transmission during a selected round number. The round number is, at least in part, based on the deficit of the flow to be inserted. One mechanism for selecting a round number is described further with reference to Figure 7. The flow ID of the current packet is then inserted into the service queue of the selected round number in operation 409. Insertion includes inserting the flow ID within the service queue associated with the selected round number and repositioning the service queue head pointer of the associated service queue. An example of flow ID insertion is described further below with reference to Figure 9.

The idle bit corresponding to the flow ID is then cleared in operation 410. For

example, a "1" value indicates that the flow is idle, while a "0" indicates that the flow is not idle. After the idle bit of an arrived packet's flow is cleared, the packet arrival process 400 ends.

Before describing examples for how to calculate a round number, it may be useful to describe how the deficit counters associated with each flow are updated since the round number selection for inserting an arrived packet's flow ID within a service queue is based on the deficit counter of the arrived packet's flow. In the illustrated embodiment, the deficit counter represents the number of rounds until the flow can transmit again. The deficit counter value may be stored within any suitable location. In the illustrated embodiment of Figure 3B, each flow location within the flow records 307 includes a deficit counter (as well as a flow ID, flow weight, credit counter, idle bit, and next flow pointer).

There are at least two situations where a deficit counter of a particular flow may be updated: when the flow goes idle and when a round number is being determined for a newly arrived packet's flow. When a flow goes idle after transmitting a packet, the corresponding deficit counter is set to the number of rounds that must elapse before the corresponding flow becomes eligible for service again. A non-zero value of the deficit counter indicates that the corresponding flow has temporarily consumed more bandwidth than its allocated share, due to the size of its most recently transmitted packet exceeding the available credit.

The deficit counter of a particular flow is periodically updated when the particular flow is idle since the flow has a new deficit value after each round number is processed. That is, as each round number is processed, the idle flow's deficit (if

any) is decreased by one. The deficit counter of an idle flow may be updated at any suitable frequency such that processing resources are conserved. For example, a deficit update process may be invoked periodically and executed as a low-priority background process, concurrently with the main scheduling processes (*e.g.*, packet arrival process 400 of Figure 4 and packet transmission completion process 1100 of Figure 11).

The deficit counter may also have to be updated when a round number is selected for inserting a newly arrived packet's flow ID since the actual deficit counter value may not accurately include all of the rounds that have passed since the last deficit update scan that was performed while the flow was idle. In other words, since the deficit counter is not updated for each round to conserve processing resources, as compared to conventional deficit round-robin schedulers, one or more rounds may have transpired between the time that the deficit counter was last updated and a round number is to be selected for the now active flow. Processes for updating the deficit counter during a round number selection process are described below in reference to Figure 7.

Figure 5 illustrates a flowchart of a process 500 of performing a deficit counter scan to update the deficit counters of idle flows in accordance with one embodiment of the present invention. During the deficit counter scan, the flows are analyzed in sequence. During each flow analysis, the corresponding deficit counter is updated if the deficit counter has a positive value (*i.e.*, the corresponding flow has a deficit) and the flow is idle.

Initially, a variable start-of-current-scan-cycle is set to the current round

number in progress at the scheduler in operation 501. The variable start-of-current-scan-cycle is used to determine in which round the deficit counter was last updated. The variable scan-flow-index is then set to zero so that it points to the first flow in operation 503. The variable scan-flow-index is an index that references the current flow that is being processed among all the pending flows. For example, scan-flow-index "0" references flow0 which is the first entry within service queues 307. All of the flows within service queues 307 may then be sequentially accessed by incrementing the scan-flow-index.

After the scan-flow-index is set to zero, the current flow (*i.e.,* first flow) is then selected via the scan-flow-index in operation 505. It is then determined whether the idle bit of the current flow is set in operation 507. If the idle bit is set, it is then determined whether the deficit counter of the current flow is greater then zero in operation 509. If it is greater then zero, the deficit counter of the current flow is updated in operation 511. The deficit counter of the current flow may be updated by decreasing the deficit counter by the number of rounds that have elapsed since the last update. An update process is described below with reference to Figure 6.

If the idle bit of the current flow is zero in operation 507 or the deficit counter of the current flow is not greater then zero in operation 509, it is then determined whether the scan-flow-index is pointing to the last flow in operation 513. This determination operation is also performed after updating the deficit counter of the current flow in operation 511. If the scan-flow-index is not pointing to the last flow, the scan-flow-index is incremented in operation 515. The deficit counter scan procedure 500 then proceeds to operation 505 so that a next flow that is referenced by

the scan-flow-index may be selected. When the scan-flow-index references the last flow in operation 513, the start-of-previous-scan-cycle is then set to the start-of-current-scan-cycle in operation 517 and the deficit counter procedure 500 ends. The start-of-previous-scan-cycle is used within the process for determining the round number for insertion of a newly arrived packet's flow ID (*i.e.*, process 408 of Figure 7), for example, to determine when the last deficit counter update occurred.

Figure 6 is a flowchart illustrating the operation 511 of Figure 5 for updating the deficit counter of the current flow during the scan cycle 500 in accordance with one embodiment of the present invention. Initially, a current offset is set to the number of elapsed rounds between the start-of-current-scan-cycle and the start-of-previous-scan-cycle in operation 601. In other words, the start-of-previous-scan-cycle is subtracted from the start-of-current-scan-cycle. This subtraction is preferably done as a modulo k operation to allow the round number to wrap around after reaching the maximum number allowed in the particular implementation.

After the offset is calculated, the deficit counter of the current flow is then decreased by the current offset in operation 603. In other words, the deficit counter is decreased by the number of rounds that have passed since the last deficit counter update. It is then determined whether the deficit counter of the current flow is less then zero in operation 605 (*i.e.*, there is no longer a deficit). If it is not less than zero, the update procedure 511 ends. If the deficit counter is less the zero, the corresponding flow has become eligible for service in the current round. The deficit counter of the current flow is then set to zero in operation 607 so that any future checks for eligibility of the current flow for service may simply compare the deficit

counter to zero. The update procedure 511 then ends.

Figure 7 illustrates the flowchart of the operation 408 of Figure 4 for selecting the round number for insertion of the flow ID of the arrived packet, when the flow queue is empty prior to the arrival of the packet, in accordance with one embodiment of the present invention. The round number is selected based on the deficit counter and the number of rounds that have elapsed since the last deficit counter update for the arrived packet's flow. The number of elapsed rounds are subtracted from the deficit counter, and the round number is then calculated by adding the deficit counter (if greater than zero) to the current round in progress.

Initially, the deficit counter of the flow corresponding to the arrived packet is examined in operation 701. It is then determined whether the deficit counter is equal to zero in operation 703. If the deficit counter is equal to zero (*i.e.*, the flow is immediately eligible for transmission), the round for insertion of the flow is selected as the current round in progress in operation 717 and the round number determination operation 408 ends.

If the deficit counter is not equal to zero (*i.e.*, the flow is not eligible for transmission), it is then determined whether the scan-flow-index is less then the flow ID corresponding to the arrived packet in operation 705. If the scan-flow-index is less then the flow ID, the current scan in progress has not yet updated the deficit counter of the flow corresponding to the arrived packet. The last update of the deficit counter occurred during the previous scan cycle of the deficit counter scan process 500. Thus, an offset is then determined as the number of rounds between the start-of-previous-scan-cycle and the current round in progress in operation 709.

If the scan-flow-index is not less then the flow ID, the current scan in progress has updated the deficit counter for the arrived packet. Thus, the offset is then determined as the number of rounds between the start-of-current-scan-cycle and the current round in progress in operation 707.

After the offset is determined, it is then subtracted from the deficit counter in operation 711. This subtraction takes into count the number of rounds that have elapsed since the deficit counter was last updated, which is illustrated and described below with reference to Figure 8. After the offset is subtracted from the deficit counter, it is then determined whether the deficit counter is greater then zero in operation 713. If the deficit counter is greater then zero (*e.g.*, the flow has to wait for a later round), a round for insertion of flow is selected as the current round in progress plus the valuc of the deficit counter in operation 715. However, if the deficit counter is not greater then zero (*e.g.*, the flow is ready for immediate transmission), the round for insertion of the flow is selected as the current round in progress in operation 717. After the round for insertion of the flow is selected, the round number determination procedure 408 ends.

Two different cases for calculating a round number may be illustrated with reference to Figure 8. Figure 8 shows a timeline 800 of rounds that were in progress, are in progress, and are scheduled for process. As shown, the current round in progress is round number 34. The round that was in progress during the start of the current scan in progress is round number 32, and the round that was in progress during the start of the previous scan is round number 25. Let's assume that when flow 15 went idle, its deficit counter had a value of 12.

In Case I, the current scan cycle has already processed flow 15. In this case, the scan cycle would have reduced the deficit counter of flow 15 by the number of elapsed rounds between the starting points of the two scan cycles, which is calculated as (32-25) = 7. Thus, the deficit counter now reads (12-7) = 5. Operation 707 of Figure 7 now calculates the difference between the current round in progress and the start of the current scan cycle, which is calculated as (34-32) = 2. Operation 711 then subtracts this difference from the deficit counter in operation 711, which calculation results in a deficit of (5-2) = 3. Operation 715 then calculates the round number in which flow 15 will become eligible for service as the current round number in progress plus the deficit, which results in (35+3) = 38.

In Case II, the current scan cycle has not reached flow 15. In this case, the deficit counter remains at 12. Operation 709 of Figure 7 calculates a difference between the current round in progress and the start of the previous scan cycle, which is calculated as (34-25) = 9. Operation 711 then subtracts this difference from the deficit counter, and the resulting deficit is (12-9) = 3. Operation 715 determines the round number in which flow 15 will become eligible for service as the current round number in progress plus the deficit, which is then calculated as (35+3) = 38, which gives the same result as Case I.

Figure 9 is flowchart illustrating the operation 409 of Figure 4 of inserting the flow ID of the arrived packet in accordance with one embodiment of the present invention. This operation generally involves rearranging the service queue head pointer for the selected round and inserting the flow ID into the service queue of the selected round. Figure 10A shows an example of a plurality of service queue head

pointers 305 and service queues 307 prior to insertion of the flow 5. As shown, each memory location within the service queue head pointers 305 corresponds to a particular round number. Each round number is associated with a starting flow ID or number, which flow is transmitted first during the associated round number. As shown, round 3 of the service queue head pointers contains a head pointer to flow 2. Thus, flow 2 is scheduled to transmit data first during round 3.

Each flow record within the service queues 307 corresponds to a particular flow. The top memory location of the service queues corresponds to flow 0; the next memory location to flow 1; etc. In the illustrated embodiment, each memory location contains either a null or a next flow pointer. A next flow pointer references the next flow for transmission for a particular round number. As shown, round 3 of the service queue head pointers 305 references flow 2 of the services queues 307, and flow 2 of the services queues 307 contains a next flow pointer for flow 3. Thus, for round 3 flow 2 is scheduled for transmission, then flow 3 is scheduled for transmission. Likewise, flow 3 of the services queues 307 contains a next flow pointer for flow 4; flow 4 contains a next flow pointer for flow 6; and flow 6 contains a next flow pointer for flow 7. Flow 7 contains a null value, thus, flow 7 represents the last transmitted flow within the service queue 307 for round 3.

Each flow of the service queue 307 is either idle or linked to an associated service queue of a particular round number of the service queue head pointers 305. As shown, flows 2, 3, 4, 6 and 7 together form a service queue for round 3. In contrast, a flow may be idle and not linked to any particular service queue. As shown, flow 5 contains a null value (*e.g.*, it does not contain a next flow pointer) and is not

referenced by any other flow. Thus, flow 5 is not associated with any other flows of a particular round number or service queue.

Figures 10A through 10C illustrate an example of implementing the process 409 of Figure 9 to insert a flow 5 within the service queue for round 3. Turning back to Figure 9, a service queue head pointer for the selected round is obtained in operation 901. In this example, the service queue head pointer corresponds to round 3 (see Figure 10A). It is then determined whether the head pointer is null in operation 903. In the illustrated example of Figure 10A, round 3 references flow 2. If the obtained service queue head pointer is null, it is set equal to the flow ID of the arrived packet in operation 905. In our example, if round 3 is null (*i.e.*, not referencing flow 2), the service queue head pointer for round 3 is set to flow 5 (not shown). A next flow pointer in the service queue corresponding the flow ID is then set to null in operation 907 and the insert flow ID procedure 409 ends. Thus, flow 5 of the service queues 307 then contains a null value (not shown).

If the head pointer is not null, the next flow pointer in the service queue corresponding to the flow ID to be inserted is then set to an old value of the head pointer for the selected round in operation 909. Figure 10B illustrated the results of this operation. The next flow pointer of flow 5 within the service queues 307 is set to flow 2, which is the current starting flow for round 3. The head pointer is then set to the flow ID of the arrived packet in operation 911. Figure 10C shows the results of operation 911. As shown in Figure 10C, the service queue head pointer for round 3 references flow 5. Flow 5 will be transmitted first for round 3 since it is referenced by the service queue head pointer 307 of round 3. Flow 2 is transmitted next for round 3

since flow 5 contains a next flow pointer to flow 2. Likewise, flows 3, 4, 6, and 7 are transmitted in sequence based on their respective next flow pointers. The insert flow ID procedure 409 then ends.

Figure 11A is a flowchart illustrating the process 1100 that occurs after a current packet (*e.g.*, an arrived packet of Figure 4) completes transmission in accordance with one embodiment of the present invention. After a packet completes transmission, a next packet is selected for transmission either from the same flow as the completed packet, from a next flow within the same service queue as the completed packet, or from a next service queue for a next round. A next packet is selected from the same flow as the completed packet unless the completed packet's flow obtains a deficit or becomes empty. If the completed packet's flow obtains a deficit, the flow is deleted from the service queue for the current round and inserted within a later round based, at least in part, on the flow's deficit. If the flow is empty, the flow is also deleted from the service queue, but the flow also becomes idle. If the flow ID corresponding to the completed packet is deleted from the service queue, the next packet for transmission is selected from a next flow of the service queue or a next service queue of a next round if the current service queue is empty.

Initially, the current packet is deleted from the packet memory in operation 1101. The size of the transmitted packet is then subtracted from a credit counter corresponding to the flow ID of the current packet in operation 1103. The credit counter is initialized, for example, to the credit allocation value (flow weight) of the corresponding flow during start up of the switch device. The credit counter value may be stored within any suitable location. For example, each flow location within the

service queues 307 of Figure 3B may include a credit counter, as well as a deficit counter, idle bit, next flow pointer, and flow ID. The credit counter is generally used to track the total sizes of transmitted packets, as compared to the credit allocation value. Each flow is assigned a credit allocation value that determines how much data may be transmitted during a single round. For example, flow 0 may have a credit allocation of 400 bytes and can thereby transmit 400 bytes per round.

The illustrated scheduling algorithms only require knowledge of the packet size when the credit counter is being updated in operation 1103. This update occurs after the entire packet is transmitted. Thus, the present invention allows scheduling of a packet without advance knowledge of the arriving packet's size, as is required by conventional scheduling algorithms.

It is then determined whether the value of the credit counter is greater then zero in operation 1105. In other words, it is determined whether the transmitted packet's size is larger than the transmitted packet's corresponding credit allocation value. If it is greater then zero (*e.g.*, the completed packet's flow does not have a deficit), it is then determined whether the packet queue for the current flow is empty in operation 1107. If it is not empty, a next packet of the current flow queue is selected for transmission from the packet memory in operation 1109 and the process 1100 ends. In other words, packets are selected for transmission from the same flow unless its credit counter indicates that the corresponding credit allocation has been exceeded or the same flow has no more packets to transmit.

If the value of the credit counter is not greater then zero via operation 1105 (*e.g.*, the completed packet has a deficit), it is then determined whether the packet

queue for the current flow is empty in operation 1111. If it is not empty, the flow ID is deleted from the service queue for the current round in operation 1113. If the flow of the completed packet is not empty, the flow needs to be reinserted within a service queue of a later round based, at least in part, on the flow's deficit. Turning to Figure 11B, after the flow ID is deleted from the service queue of the current round in operation 1113, a round number is calculated for the next packet of the current flow in operation 1121. The round number is calculated by increasing the credit counter of the current flow by the flow's credit allocation until the credit counter becomes greater than zero. The round number is then calculated as the amount that was used to increase the credit counter over zero (*e.g.*, the flow no longer has a deficit at this round number). A round number calculation process 1121 is described further below in reference to Figure 13. The current flow ID is then inserted into the service queue at the head of the calculated round in operation 1123.

After the current flow ID is inserted via operation 1123, the next flow pointer for the current round is obtained from the service queue in operation 1125. It is then determined whether the next flow pointer for the current round is empty in operation 1127. If it is not empty, the first packet in the packet queue corresponding to the next flow is then selected for transmission in operation 1129. In other words, if a flow exceeds its credit allocation value, packets from a next flow or next round are then selected for transmission. The process 1100 then ends.

If the next flow pointer for the current round is empty via operation 1127, the current round is incremented (e.g., by a modulo k operation) in operation 1131. A null value for the next flow pointer indicates that there are no more packets to

transmit for the current round. For example, no flows are eligible for transmission during the current round. Thus, the next round is obtained in operation 1131 when the current round has no more packets to transmit. It is then determined whether the head pointer corresponding to the new current round is null in operation 1133. If it is null, the current round is again incremented in operation 1131. In other words, the round number is incremented until a nonempty round is found. Of course, if there are a limited number of round numbers, the round numbers are incremented until the maximum round number is reached and then the first round number (*i.e.*, round number 0) is reused. When the current round has a head pointer that is not null, the first packet in the packet queue corresponding to the flow identified by the head pointer is selected for transmission in operation 1135. The process 1100 ends.

Turning back to Figure 11A, if the packet queue for the current flow is empty via operation 1107 or 1111, the flow ID is deleted from the service queue of the current round in operation 1115. The deficit counter for the current flow is then set in operation 1117. Setting the deficit counter basically includes setting the deficit counter so that a correct round number may later be determined for inserting a later arriving packet from the deleted flow. One embodiment of the process 1117 for setting the deficit counter is further described below in reference to Figure 12.

The idle bit of the current flow is also set to one to indicate that the current flow is now idle in operation 1119. The idle bit is analyzed during the scan routine for updating deficit counters while a particular flow is idle (*e.g.*, as illustrated in the scan routine 500 of Figure 5). The process 1100 the proceeds to "B" of Figure 11B, and a next flow pointer for the current round is obtained from the service queue in

operation 1125 of Figure 11B.

Figure 12 is flowchart illustrating the operation 1117 of Figure 11 for setting the deficit counter of a flow when the flow goes idle in accordance with one embodiment of the present invention. The deficit counter is set such that it reflects the round number in which the flow again becomes eligible for transmission. The deficit counter of the flow is initialized to zero in operation 1201. It is then determined whether the credit counter of the flow is greater then zero in operation 1203. If the credit counter is not greater than zero, the credit counter is increased by the credit allocation value in operation 1213 and the deficit counter is incremented in operation 1215 until the credit counter is greater than zero. Repeating operations 1213 and 1215 results in a deficit counter that indicates how many times the credit counter must be increased by the credit allocation value before the credit counter becomes a positive value. Since the corresponding flow is allowed to transmit an amount of data equal to the credit allocation during each round, the deficit counter now indicates the number of rounds required until there is no longer a deficit, as indicated by the credit counter.

When the credit counter is greater then zero, the deficit counter indicates how many rounds must elapse before the corresponding flow becomes eligible for transmission. However, the deficit counter has been calculated with respect to the current round in progress. If the process for updating the deficit counter while the flow is idle is not performed relative to the current round (as the scan routine 500 of Figure 5), an offset may be calculated to reflect this discrepancy.

Thus, it is then determined whether the scan-flow-index is less than the flow

ID of the current flow in operation 1205. In other words, has the current flow's deficit counter been updated already. If the scan-flow index is less that the flow ID (*i.e.*, the flow has not been processed), the offset is determined as the number of rounds between the start of the previous scan cycle and the current round in progress in operation 1209. If the scan-flow-index is not less than the flow ID of the current flow (*i.e.*, the flow has not been processed), the offset is determined as the number of rounds between the start of the current scan cycle and the current round in progress in operation 1207. After the offset is determined via operation 1209 or 1207, the offset is then added to the deficit counter in operation 1211 and the operation for setting the deficit counter 1117 ends.

Referring back to the example of Figure 8, assume that the queue of flow 15 becomes idle after transmitting a packet, and its actual deficit is 3. Since the current round in progress is 34, flow 15 will become eligible for service again in round number 38 (*i.e.*, after the deficit has been used up within rounds 35, 36, and 37). When the processing operations in Figure 12 are performed, operation 1205 determines that flow 15 has not yet been updated during the current scan cycle. Hence, the adjustment to its deficit is calculated as the number of elapsed rounds between the start of the previous scan cycle and the current round in progress in operation 1209 (*i.e.*, 34-25 = 9). Therefore the value of the deficit counter computed by the flowchart 7 is (3+9) = 12.

Now let us examine how the round number is determined on a packet arrival for insertion of a flow ID into the service queues when it has been idle. The operations in flowchart 7 must now determine the round number of the service queue

in which flow 15 is to be inserted. As described above, this calculation will follow one of the two paths in Figure 7, depending on how far the current scan cycle has progressed. In a case where the current scan cycle has already processed flow 15 (Case I), the scan cycle reduced the deficit counter of flow 15 by the number of elapsed rounds between the starting points of the two scan cycles, that is (32-25) = 7. Thus, the deficit counter now reads (12-7) = 5. In operation 707 of Figure 7, the difference between the current round in progress and the start of the current scan cycle (34-32 = 2) is subtracted from the deficit counter in operations 707 and 711, and the resulting deficit is (5-2) = 3. The round number in which flow 15 will become eligible for service is calculated as 38 (*i.e.*, 35+3) in operation 715, which is the desired result.

In the case where the current scan cycle has not reached flow 15 (Case II), the deficit counter remains at 12. The difference between the round in progress and the start of the previous scan cycle (34-25 = 9) is subtracted from the deficit counter in operations 709 and 711, and the resulting deficit is 3 (*i.e.*, 12-9). The round number in which flow 15 will become eligible for service is calculated as 38 (*i.e.*, 35+3) in operation 715, which is again the desired result.

Figure 13 is flowchart illustrating the operation 1121 of Figure 11B of calculating the round number in accordance of one embodiment of the present invention. The round number may then be used to reinsert the deleted flow in a later round. The round number is calculated by incrementing the round number value and increasing the credit counter value by the credit allocation value until a positive credit counter value is reached. The round number value then represents the calculated

round number value.

Initially, the next round is set equal to the current round in operation 1301. The value of the credit counter of the flow is then increased by the credit allocation of the flow in operation 1303. The next round is then incremented in operation 1305. It is then determined whether the value of the credit counter is greater then zero in operation 1307. If it is greater then zero, the round number calculation 1121 ends since the round number in which there is no longer a deficit has been found. If the value of the credit counter is not greater then zero, the value of the credit counter of the flow is increased again by the credit allocation of the flow in operation 1303. The credit counter continues to be increased in operation 1303 and the next round continues to be incremented in operation 1305 until the credit counter is greater then zero.

Although the foregoing invention has been described in some detail for purposes of clarity of understanding, it will be apparent that certain changes and modifications may be practiced within the scope of the appended claims. It should be noted that there are many alternative ways of implementing both the process and apparatus of the present invention. For example, although each service queue head pointer is described as referencing a flow ID of a first flow, of course, each service queue head pointer may directly reference the first packet of the first flow. Accordingly, the present embodiments are to be considered as illustrative and not restrictive, and the invention is not to be limited to the details given herein, but may be modified within the scope and equivalents of the appended claims.

What is claimed is:

CLAIMS

1. A method for scheduling a data portion for transmission through an output port, wherein the data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, the data portion being associated with a flow, the method comprising:

selecting a round number from the plurality of round numbers for transmitting the data portion through the output port; and

associating the flow associated with the first data portion with the selected round number such that the first data portion is automatically transmitted during the selected round number without first calculating a deficit or surplus for the flow associated with the first data portion.

2. A method as recited in claim 1, wherein when the flow associated with the first data portion contains other data portions or the flow does not have a deficit, the round number is selected as a round number that is currently transmitting data.

3. A method as recited in claim 2, wherein when the flow associated with the first data portion is empty and has a deficit, the round number is selected as a later round number in which the first data portion becomes eligible for transmission.

4. A method as recited in claim 1, further comprising:

after the first data portion is transmitted during the selected round number and if the associated flow is not empty but has obtained a deficit, selecting a second round number from the plurality of round numbers for transmitting a next data portion associated with the flow through the output port, the second selected round number

transmitting data after the first selected round number; and

associating the flow with the second selected round number such that the second data portion is automatically transmitted during the second selected round number without first calculating a deficit or surplus for the flow associated with the second data portion.

5. A method as recited in claim 4, further comprising after the first data portion is transmitted during the selected round number and if the associated flow is empty, periodically updating a deficit counter to track the flow's deficit while the flow is idle, the updating occurring less frequently than a frequency between round numbers.

6. A method for maintaining a data structure that is used to schedule transmission of a plurality of data portions during a plurality of sequentially numbered rounds, the data structure including a plurality of service queues that are each associated with a one of the round numbers, each of the service queues also being associated with one or more flows, each associated flow being eligible for transmission when its associated round number is reached, the method comprising:

receiving a current data portion associated with a current flow that is associated with a current deficit counter and a current credit counter;

maintaining the current credit counter to indicate whether the current flow has a deficit and is ineligible to transmit;

maintaining the current deficit counter to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission; and

when the current data portion's associated flow is not associated with one of the service queues, determining a round number in which the current flow is eligible for transmission based on the current deficit counter and associating the current flow with the service queue of the determined round number.

7. A method as recited in claim 6, wherein the current flow is associated with the service queue of the determined round number by having a service queue head pointer associated with the determined round number reference the current flow.

8. A method as recited in claim 7, wherein the current flow is further associated with the service queue of the determined round number by having a next flow pointer of the current flow reference a next flow within the service queue of the determined round number if the service queue of the determined round was not empty prior to being associated with the current flow and references a null value if the service queue was empty.

9. A method as recited in claim 6, wherein the credit counter is maintained such that it represents a difference between a total number of bytes that have been transmitted for the current flow and a total credit allocation indicating how many bytes are assigned to the current flow.

10. A method as recited in claim 6, further comprising selecting a next data portion for transmission based at least on the current credit counter of the current flow after the current data portion is transmitted.

11. A method as recited in claim 10, wherein another data portion from the current flow is selected as the next data portion when the current credit counter

indicates that there is not a deficit and the current flow is not empty.

12. A method as recited in claim 10, wherein the next data portion is selected from a next flow within the service queue of the determined round number when the current credit counter indicates that there is a deficit or the current flow is empty and the service queue of the determined round number is not empty.

13. A method as recited in claim 12, further comprising determining a next round number in which the current flow is eligible for transmission based on the current credit counter and associating the current flow with the service queue of the determined next round number when the current credit counter indicates that there is a deficit or the current flow is empty and the service queue of the determined round number is not empty.

14. A method as recited in claim 10, wherein the next data portion is selected from a next flow within the service queue of a next round number when the service queue of the determined round number is empty.

15. A method as recited in claim 10, further comprising periodically decreasing the current deficit counter by how many rounds have passed since a previous decrease to the current deficit counter while the current flow is empty and the current deficit counter indicates that the current flow has a deficit.

16. A method as recited in claim 15, further comprising setting the current deficit counter equal to a number of rounds until the current credit counter indicates that there is not a deficit plus an offset equal to a number of rounds between the last decrease to the current deficit counter and a current round in progress when the

current flow if the current flow is empty.

17. A method as recited in claim 15, wherein the current deficit counter is decreased in a background, low priority process.

18. A method as recited in claim 15, wherein the round number of the current flow in which the current flow is eligible for transmission is determined based on the current deficit counter minus how many rounds have passed since the last decrease to the current deficit counter.

19. An apparatus for scheduling a data portion for transmission through an output port, wherein the data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, the data portion being associated with a flow, the apparatus comprising:

a storage device arranged to receive the data portion and a database arranged to associate a round number with a flow in which data portions from the flow may be transmitted;

a scheduler arranged to select a round number from the plurality of round numbers for transmitting the data portion through the output port and to assign the flow associated with the first data portion to the selected round number via the database; and

a transmitter arranged to automatically transmit the data portion during the selected round number without first calculating a deficit or surplus for the flow associated with the first data portion.

20. An apparatus for maintaining a data structure that may be used to schedule transmission of a plurality of data portions during a plurality of sequentially

numbered rounds, the data structure including a plurality of service queues that are each associated with a one of the round numbers, each of the service queues also being associated with one or more flows, each associated flow being eligible for transmission when its associated round number is reached, the apparatus comprising:

a storage device arranged to receive data portions and the structure;

a scheduler arranged to receive a current data portion associated with a current flow associated with a current deficit counter and a current credit counter, maintain the current credit counter to indicate whether the current flow has a deficit and is ineligible to transmit, maintain the current deficit counter to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission, and if the current data portion's associated flow is not associated with one of the service queues, determining a round number in which the current flow is eligible for transmission based on the current deficit counter and associating the current flow with the service queue of the determined round number.

21. A computer program product arranged to cause a computer to schedule a data portion for transmission through an output port, wherein the data portion is transmitted during one or more of a plurality of rounds that are ordered in sequential numbers, the data portion being associated with a flow, the computer program product comprising:

computer code that selects a round number from the plurality of round numbers for transmitting the data portion through the output port; and

computer code that associates the flow associated with the first data portion with the selected round number such that the first data portion is automatically transmitted during the selected round number without first calculating a deficit or

surplus for the flow associated with the first data portion; and

a computer readable medium that stores the computer codes.

22. A computer program product arranged to cause a computer to maintain a data structure that may be used to schedule transmission of a plurality of data portions during a plurality of sequentially numbered rounds, the data structure including a plurality of service queues that are each associated with a one of the round numbers, each of the service queues also being associated with one or more flows, each associated flow being eligible for transmission when its associated round number is reached, the computer program product comprising:

computer code that receives a current data portion associated with a current flow associated with a current deficit counter and a current credit counter;

computer code that maintains the current credit counter to indicate whether the current flow has a deficit and is ineligible to transmit;

computer code that maintains the current deficit counter to approximate how many rounds must pass before the current flow is eligible for transmission when the current flow is ineligible for transmission; and

computer code that determines a round number in which the current flow is eligible for transmission based on the current deficit counter and associating the current flow with the service queue of the determined round number when the current data portion's associated flow is not associated with one of the service queues; and

a computer readable medium that stores the computer codes.

23. A data structure for scheduling transmission of a plurality of data portions during a plurality of sequentially numbered rounds, the data structure

comprising:

a plurality of service queues that are associated with one or more flows; and

a plurality of service queue references that are each associated with one of the round numbers and a one of the service queues, the associated flows of each service queue being eligible for transmission when its associated round number is reached.

24. A data structure as recited in claim 23, wherein the associated flows of a selected service queue are linked together.

25. A data structure as recited in claim 23, wherein each service queue reference is in the form of a service queue head pointer that references a first flow within the associated service queue, the first flow being scheduled to transmit data first for the associated round.

26. A data structure as recited in claim 23, wherein each referenced flow includes a flow ID corresponding to the referenced flow and a next flow pointer that references a next flow within the service queue associated with the referenced flow or a null value if there is not a next flow.

27. A data structure as recited in claim 23, further comprising a packet memory for storing data portions that associated with one of the referenced flows.

28. A data structure as recited in claim 27, further comprising a plurality of per flow head pointers that each reference a first data portion of the flow associated with the first data portion.

29. A data structure as recited in claim 27, further comprising a plurality of per flow tail pointers that each reference a last data portion of the flow associated with

the first data portion.

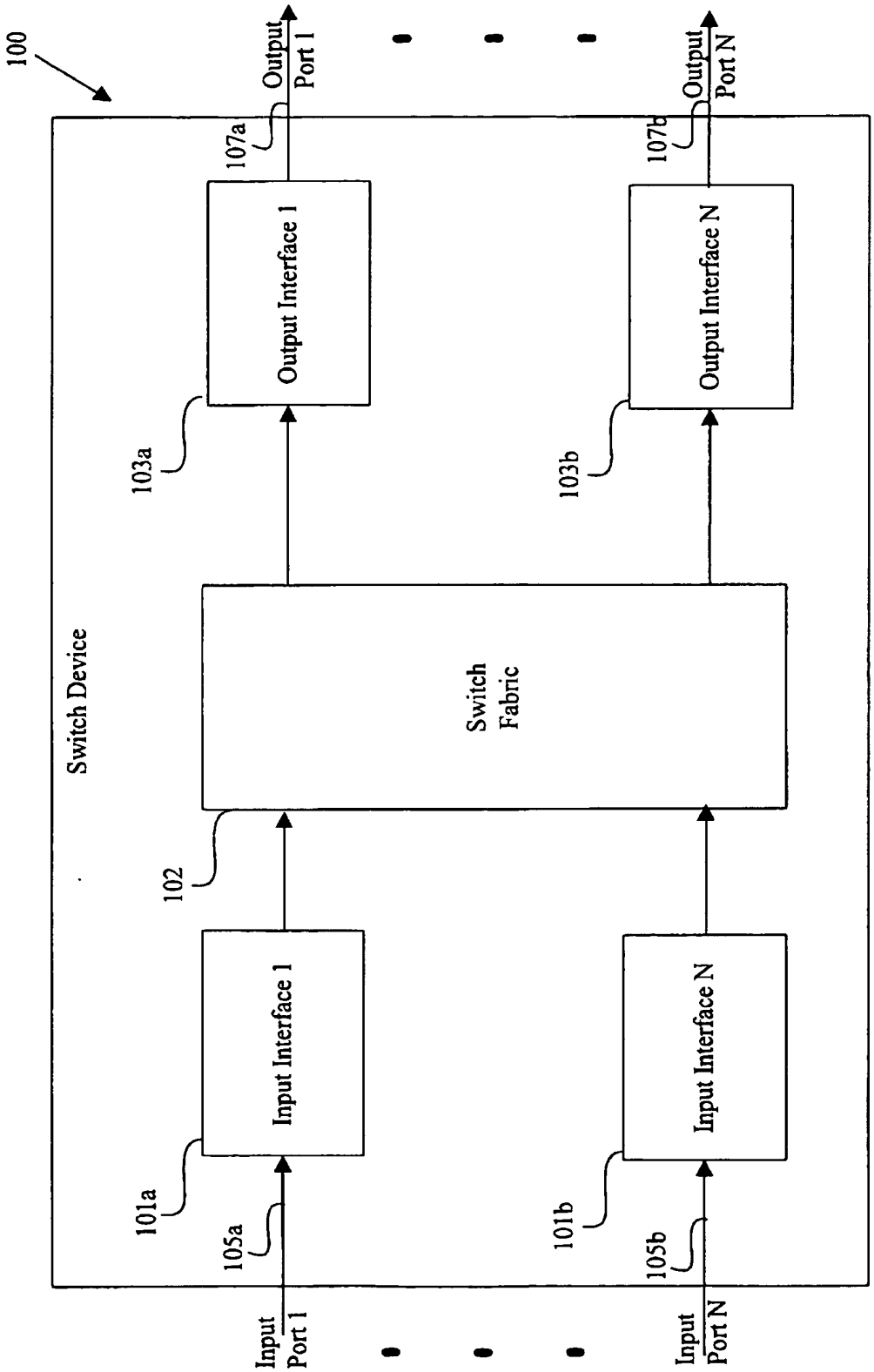

100
Switch Device
Input Port 1
Input Port N
Output Port 1
Output Port N
105a
105b
101a
101b
Input Interface 1
Input Interface N
102
Switch Fabric
103a
103b
107a
107b
Output Interface 1
Output Interface N

Figure 1

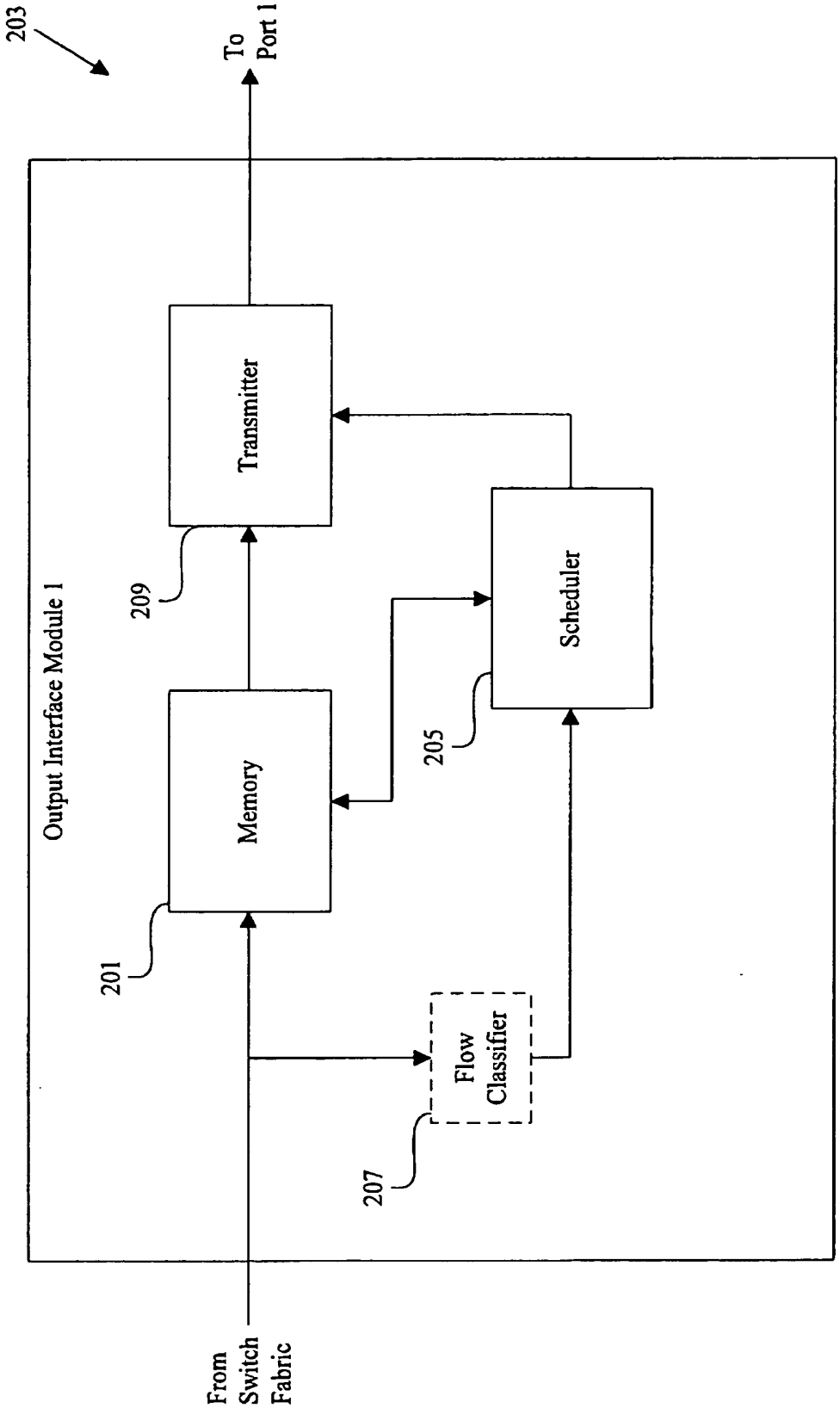

203
To Port 1
Output Interface Module 1
Transmitter
Scheduler
Memory
Flow Classifier
209
205
201
207
From Switch Fabric

Figure 2

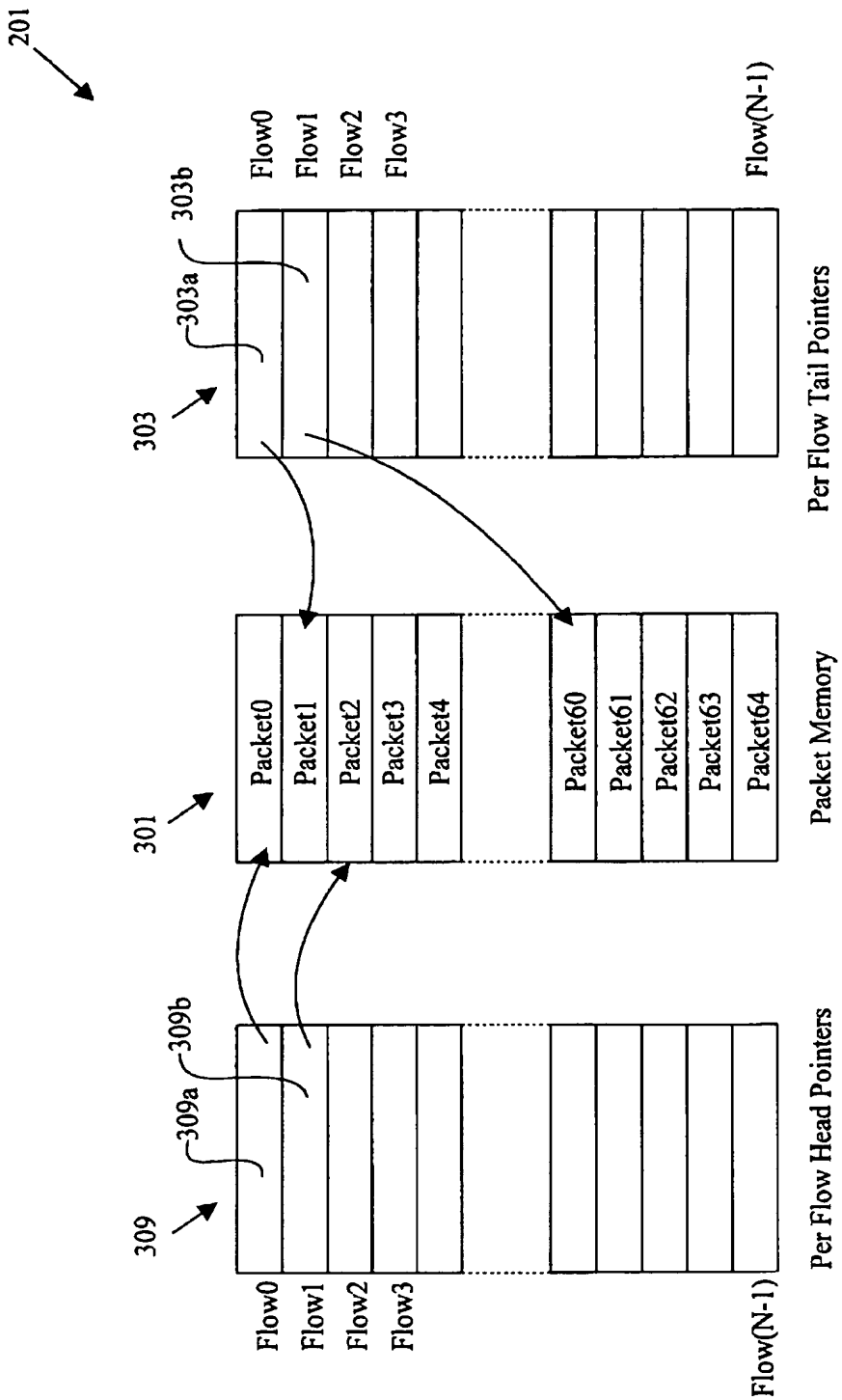

201
303
303a
303b
Flow0
Flow1
Flow2
Flow3
Flow(N-1)
Per Flow Tail Pointers
301
Packet0
Packet1
Packet2
Packet3
Packet4
Packet60
Packet61
Packet62
Packet63
Packet64
Packet Memory
309
309a
309b
Per Flow Head Pointers
Flow0
Flow1
Flow2
Flow3
Flow(N-1)

Figure 3A

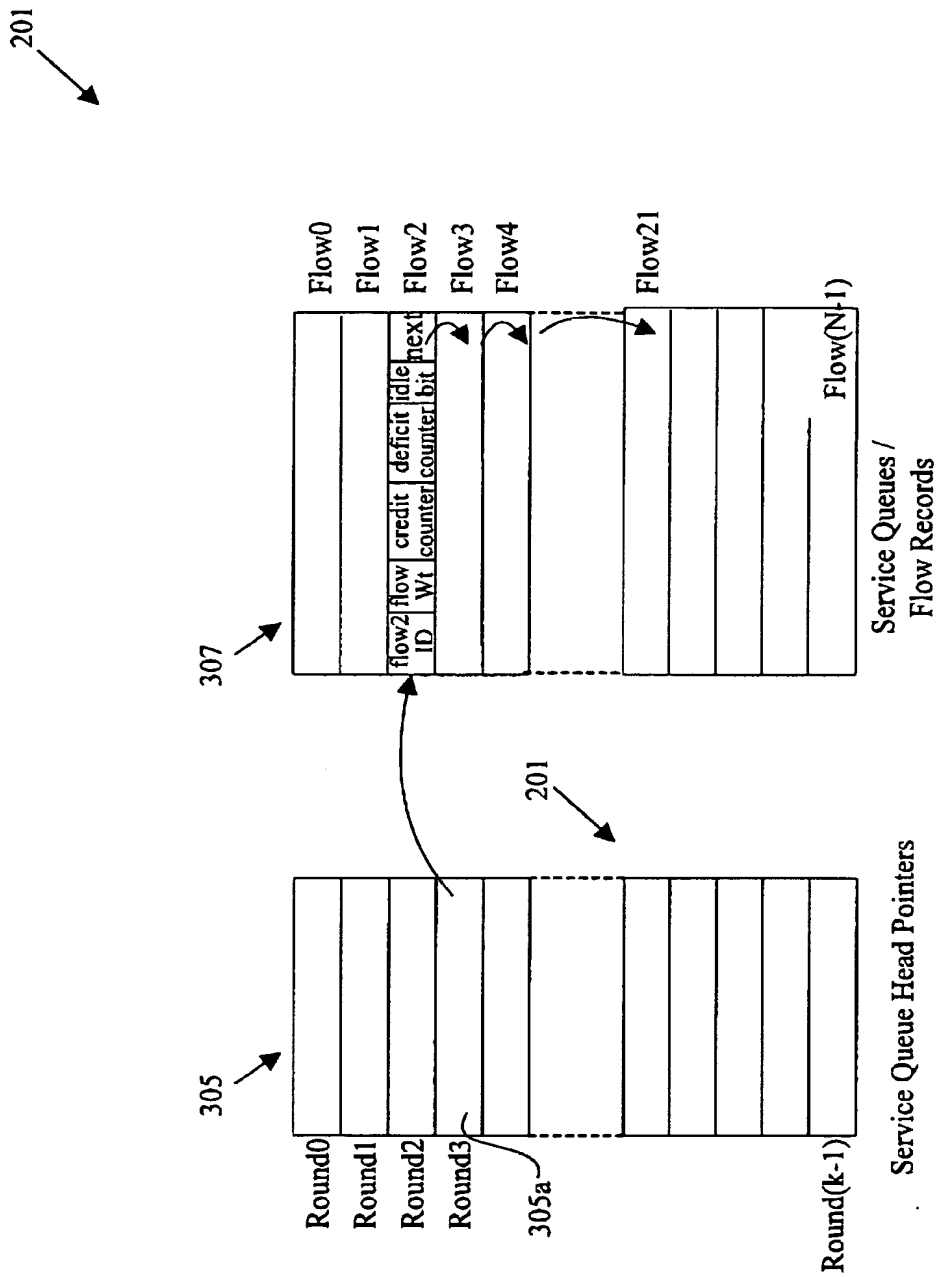

201
Flow0
Flow1
Flow2
Flow3
Flow4
Flow21
Flow(N-1)
flow2 ID
flow Wt
credit counter
deficit counter
idle bit
next
307
Service Queues / Flow Records
201
305
Round0
Round1
Round2
Round3
305a
Round(k-1)
Service Queue Head Pointers

Figure 3B

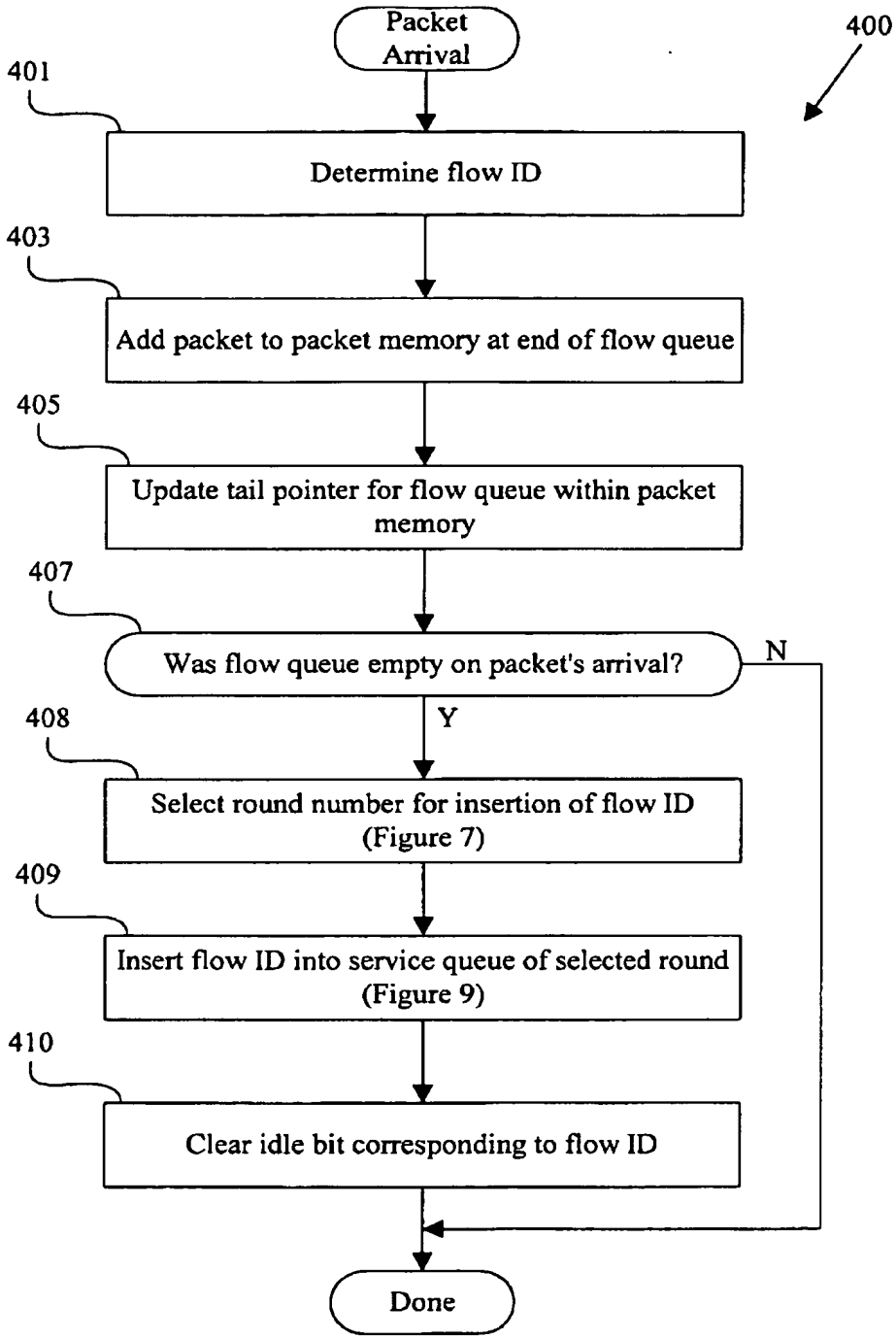

Packet Arrival
400
401
Determine flow ID
403
Add packet to packet memory at end of flow queue
405
Update tail pointer for flow queue within packet memory
407
Was flow queue empty on packet's arrival?
N
Y
408
Select round number for insertion of flow ID (Figure 7)
409
Insert flow ID into service queue of selected round (Figure 9)
410
Clear idle bit corresponding to flow ID
Done

Figure 4

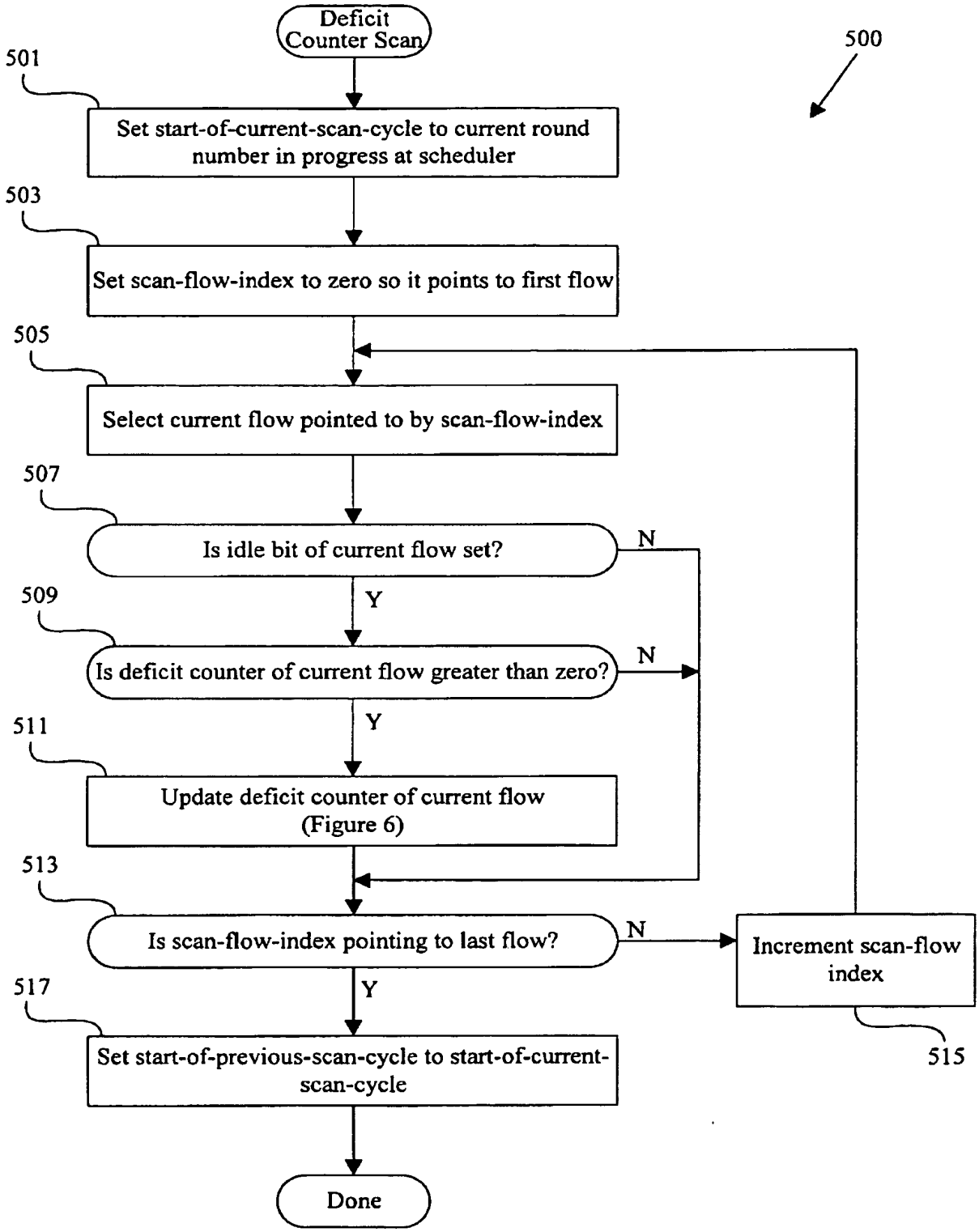

Deficit Counter Scan
500
501
Set start-of-current-scan-cycle to current round number in progress at scheduler
503
Set scan-flow-index to zero so it points to first flow
505
Select current flow pointed to by scan-flow-index
507
Is idle bit of current flow set?
N
Y
509
Is deficit counter of current flow greater than zero?
N
Y
511
Update deficit counter of current flow (Figure 6)
513
Is scan-flow-index pointing to last flow?
N
Increment scan-flow index
515
Y
517
Set start-of-previous-scan-cycle to start-of-current-scan-cycle
Done

Figure 5

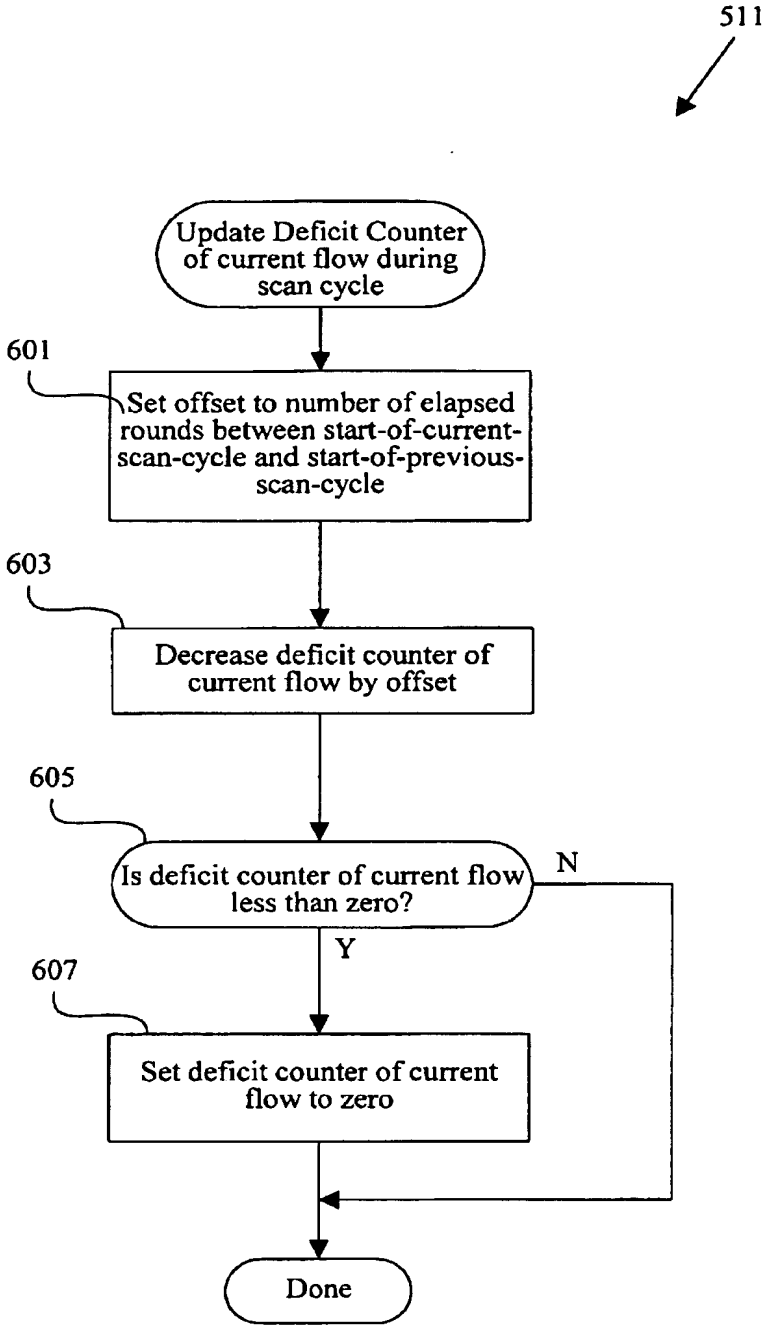

511
Update Deficit Counter of current flow during scan cycle
601
Set offset to number of elapsed rounds between start-of-current-scan-cycle and start-of-previous-scan-cycle
603
Decrease deficit counter of current flow by offset
605
Is deficit counter of current flow less than zero?
N
Y
607
Set deficit counter of current flow to zero
Done

Figure 6

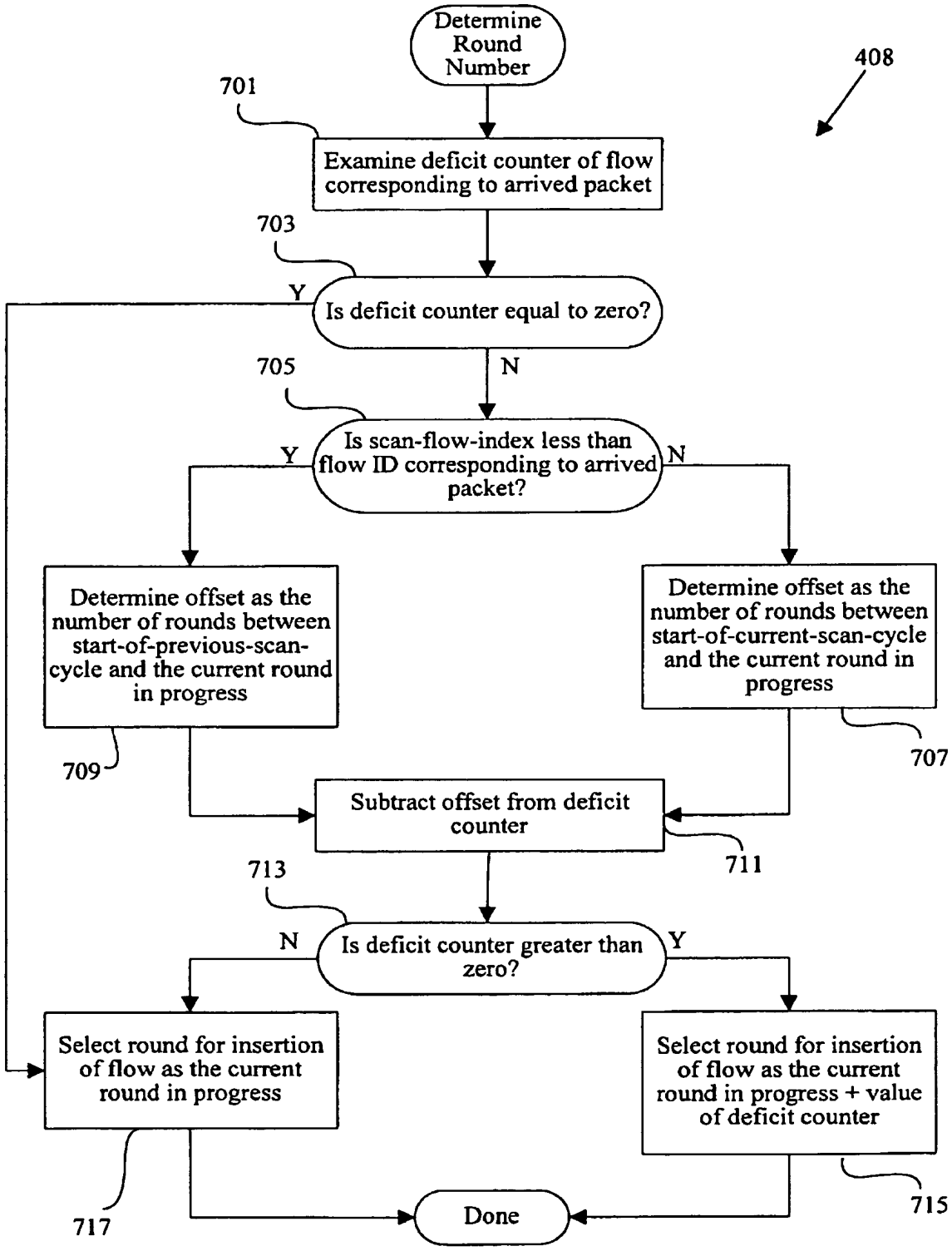

Determine Round Number
408
701
Examine deficit counter of flow corresponding to arrived packet
703
Y
Is deficit counter equal to zero?
N
705
Is scan-flow-index less than flow ID corresponding to arrived packet?
Y
N
Determine offset as the number of rounds between start-of-previous-scan-cycle and the current round in progress
Determine offset as the number of rounds between start-of-current-scan-cycle and the current round in progress
709
707
Subtract offset from deficit counter
711
713
N
Is deficit counter greater than zero?
Y
Select round for insertion of flow as the current round in progress
Select round for insertion of flow as the current round in progress + value of deficit counter
717
715
Done

Figure 7

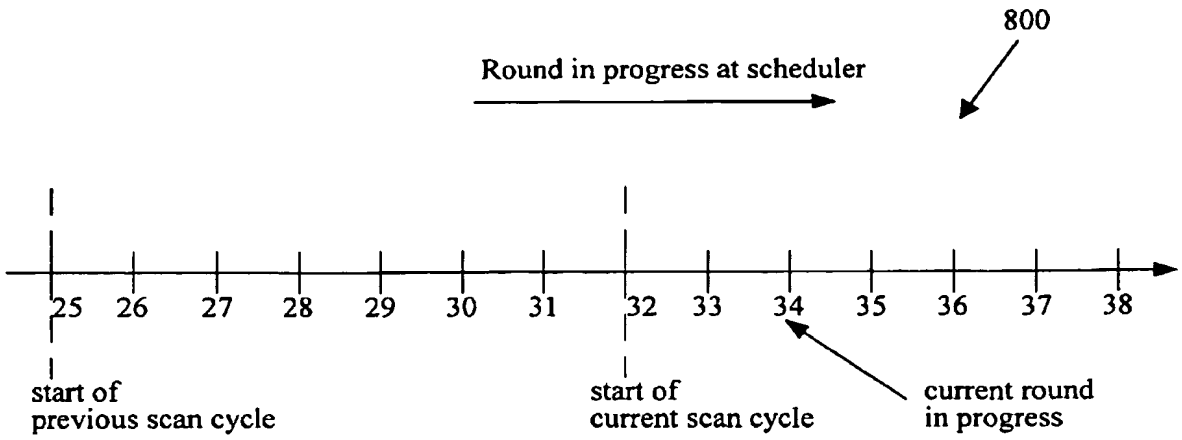


Figure 8

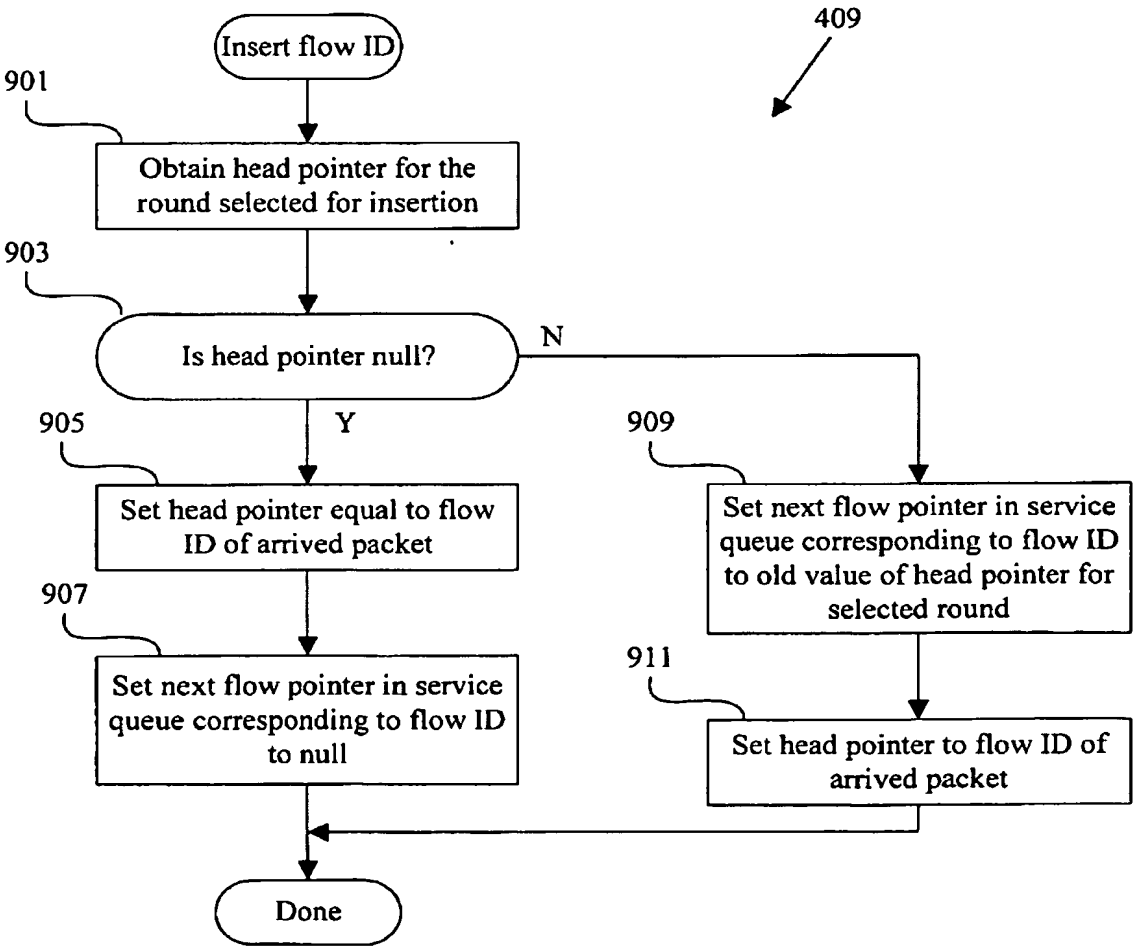


Figure 9

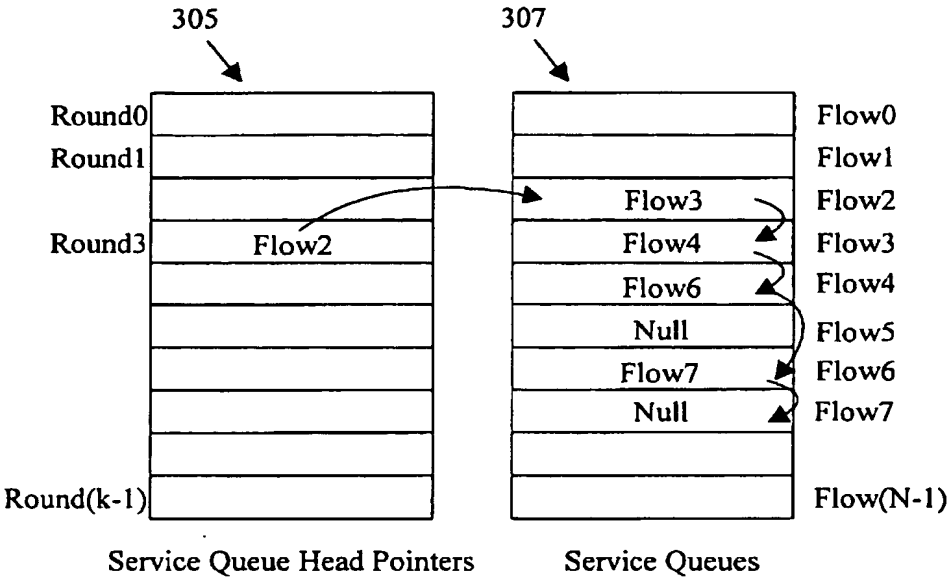


Figure 10A
(Prior to inserting flow ID 5 of Fig. 9)

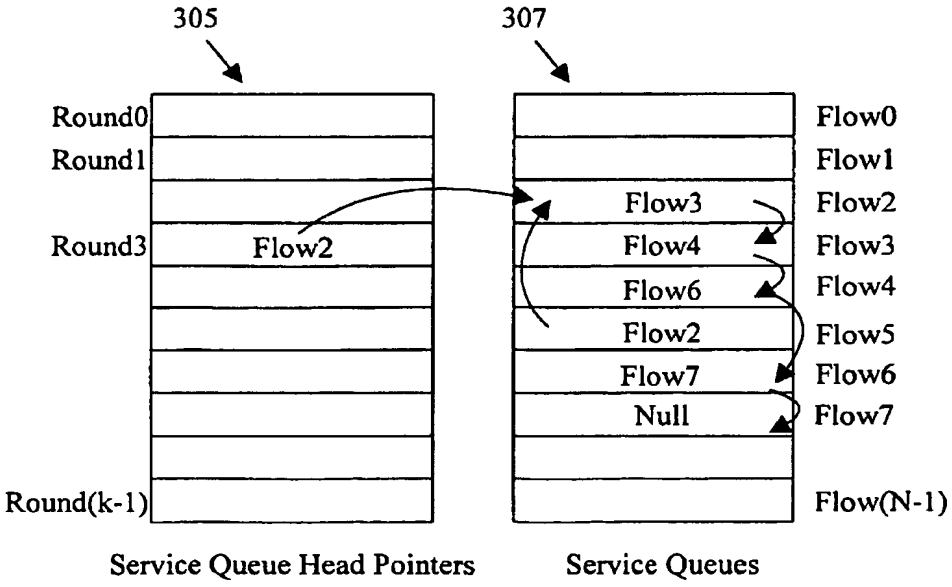


Figure 10B
(After 909 of Fig. 9)

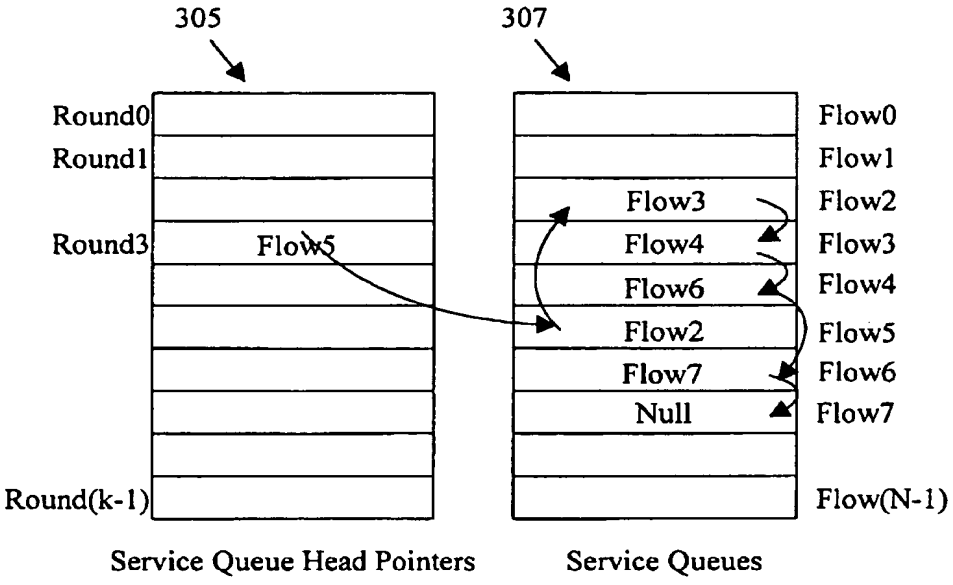


Figure 10C
(After 911 of Fig. 9)

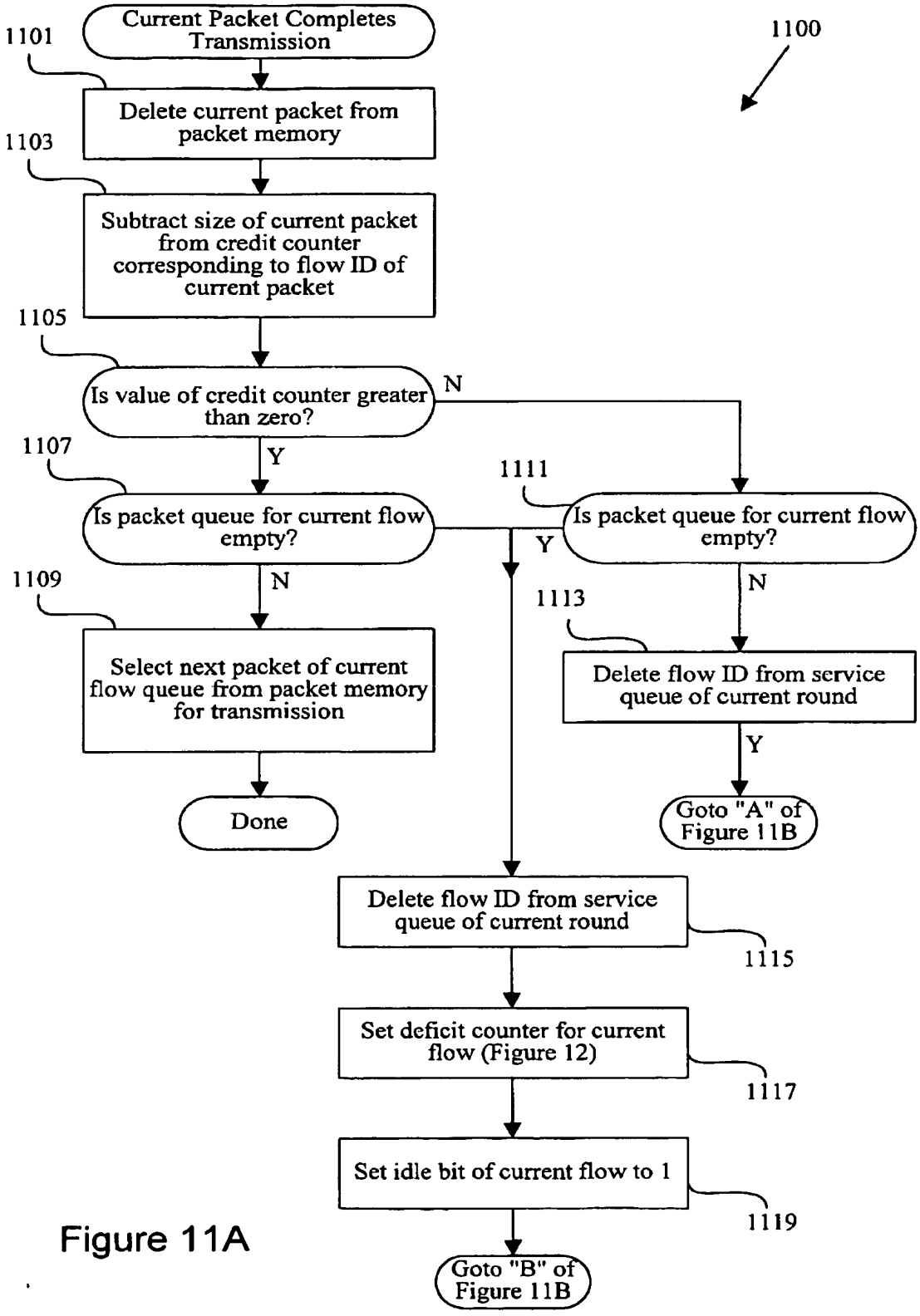

1100
1101
Current Packet Completes Transmission
Delete current packet from packet memory
1103
Subtract size of current packet from credit counter corresponding to flow ID of current packet
1105
Is value of credit counter greater than zero?
N
Y
1107
Is packet queue for current flow empty?
Y
1111
Is packet queue for current flow empty?
N
1109
Select next packet of current flow queue from packet memory for transmission
Done
N
1113
Delete flow ID from service queue of current round
Y
Goto "A" of Figure 11B
Delete flow ID from service queue of current round
1115
Set deficit counter for current flow (Figure 12)
1117
Set idle bit of current flow to 1
1119
Goto "B" of Figure 11B

Figure 11A

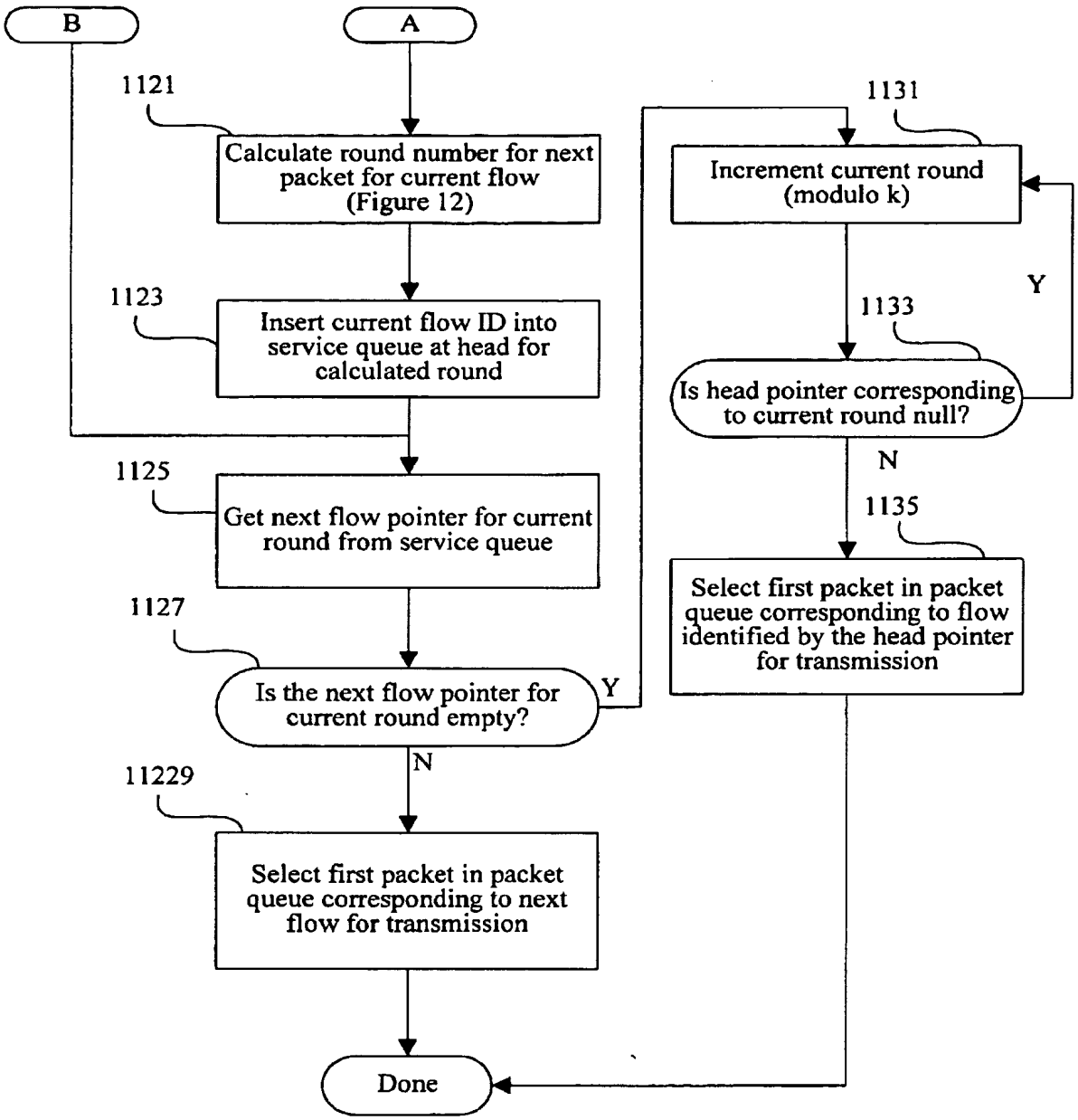

B
A
1121
Calculate round number for next packet for current flow (Figure 12)
1123
Insert current flow ID into service queue at head for calculated round
1125
Get next flow pointer for current round from service queue
1127
Is the next flow pointer for current round empty?
Y
N
11229
Select first packet in packet queue corresponding to next flow for transmission
1131
Increment current round (modulo k)
1133
Is head pointer corresponding to current round null?
Y
N
1135
Select first packet in packet queue corresponding to flow identified by the head pointer for transmission
Done

Figure 11B

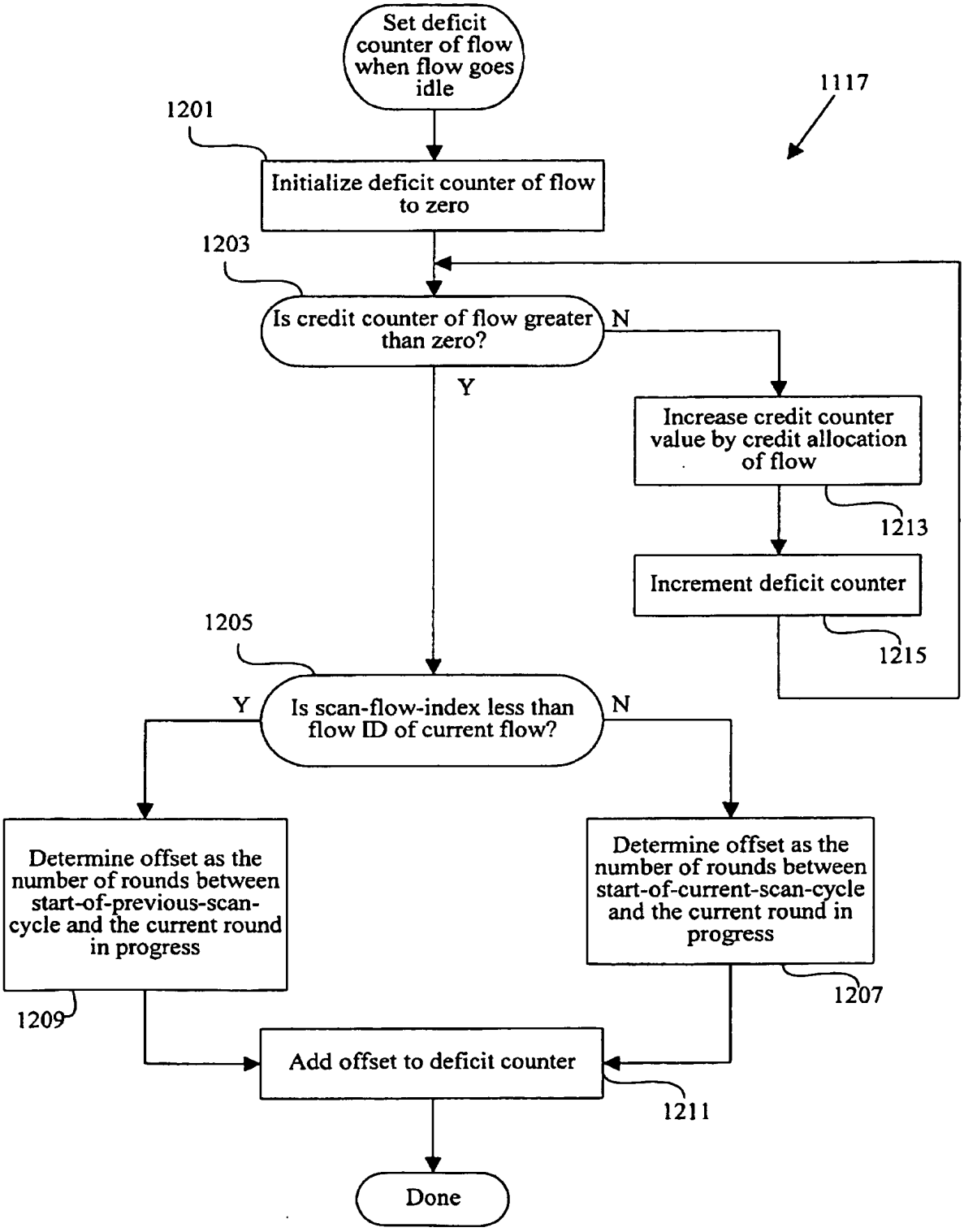

Set deficit counter of flow when flow goes idle
1117
1201
Initialize deficit counter of flow to zero
1203
Is credit counter of flow greater than zero?
N
Y
Increase credit counter value by credit allocation of flow
1213
Increment deficit counter
1215
1205
Is scan-flow-index less than flow ID of current flow?
Y
N
Determine offset as the number of rounds between start-of-previous-scan-cycle and the current round in progress
1209
Determine offset as the number of rounds between start-of-current-scan-cycle and the current round in progress
1207
Add offset to deficit counter
1211
Done

Figure 12

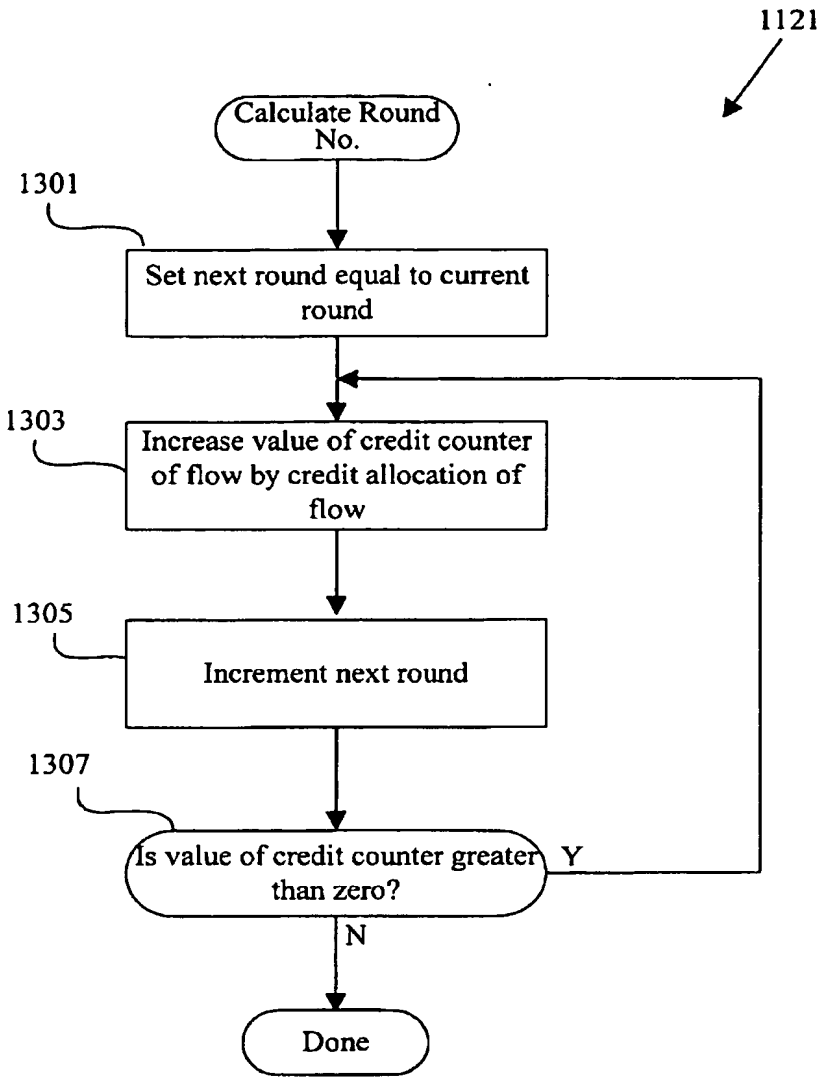

1121
Calculate Round No.
1301
Set next round equal to current round
1303
Increase value of credit counter of flow by credit allocation of flow
1305
Increment next round
1307
Is value of credit counter greater than zero?
Y
N
Done

Figure 13

INTERNATIONAL SEARCH REPORT

Int[illegible]nal Application No
PCT/US 00/28370

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L29/06 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	ALTINTAS O ET AL: "URGENCY-BASED ROUND ROBIN: A NEW SCHEDULING DISCIPLINE FOR MULTISERVICE PACKET SWITCHING NETWORKS" IEICE TRANSACTIONS ON COMMUNICATIONS,JP,INSTITUTE OF ELECTRONICS INFORMATION AND COMM. ENG. TOKYO, vol. E81-B, no. 11, 1 November 1998 (1998-11-01), pages 2013-2021, XP000790228 ISSN: 0916-8516	1,2,19, 21,23, 25,27
A	abstract figure 2 page 2014, column 1, paragraph 3 -column 2 page 2015, column 2, line 1 - line 25	6,19,20, 22
	--- -/--	

[X] Further documents are listed in the continuation of box C.

[ ] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
9 February 2001	15/02/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Lamadie, S

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Inte nal Application No
PCI/US 00/28370

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	SHREEDHAR M ET AL: "EFFICIENT FAIR QUEUING USING DEFICIT ROUND ROBIN" COMPUTER COMMUNICATIONS REVIEW,US,ASSOCIATION FOR COMPUTING MACHINERY. NEW YORK, vol. 25, no. 4, 1 October 1995 (1995-10-01), pages 231-242, XP000541659 ISSN: 0146-4833 page 233, column 2, paragraph 3 -page 235, column 1 figures 2,3	1,6, 19-23